Forschungsbericht 2016-14

Reduced-order modeling for steady transonic flows via manifold learning

Thomas Franz

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Aerodynamik und
Strömungstechnik
Braunschweig

114 Seiten
37 Bilder
8 Tabellen
95 Literaturstellen

DLR Deutsches Zentrum
für Luft- und Raumfahrt

Reduced-order modeling for steady transonic flows via manifold learning

Von der
Carl-Friedrich-Gauß-Fakultät
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines
Doktoringenieurs (Dr.-Ing.)

genehmigte Dissertation

von
Thomas Franz
geboren am 12.09.1984
in Lippstadt.

Eingereicht am:	23.10.2015
Disputation am:	26.02.2016

Referent:	Priv.-Doz. Dr. Ralf Zimmermann
1. Koreferent:	Prof. Dr. Matthias Bollhöfer
2. Koreferent:	Prof. Dr. Volker Schulz
Vorsitzender:	Prof. Dr. Dirk Langemann

2016

Abstract

This thesis presents novel parametric reduced-order models (ROMs) based on manifold learning (ML) for use in steady transonic aerodynamic applications.

The main objective of this work is to derive efficient ROMs that exploit the low-dimensional manifold of the underlying computational fluid dynamics (CFD) solutions to ensure an improved treatment of the nonlinearities involved in varying the inflow conditions and geometric parameters. In particular, an improved prediction of shocks should be obtained. The reduced-order representation of the underlying CFD data is derived using the ML method Isomap, which is extended to aerodynamic applications.

In order to develop a ROM that has the ability to predict approximate CFD solutions at untried parameter combinations, Isomap is coupled with an interpolation method, referred to as Isomap+I, to allow for variations in parameters like the angle of attack or the Mach number. Furthermore, an approximate local inverse mapping from the reduced-order representation to the full CFD solution space is introduced. In addition, the Isomap+I based predictions are further improved by optimizing the corresponding flux residual of the CFD solver used.

The low-dimensional representation of the solution manifold discovered by Isomap is also exploited to develop an adaptive sampling strategy. The goal of the method is to obtain a homogenously distributed sampling of the data manifold, which leads to a better description of the underlying manifold and eventually to a more accurate ROM.

The proposed Isomap based ROMs along with the adaptive sampling strategy are successfully applied to predict CFD solutions of the NACA 64A010 airfoil and to a fuselage-wing configuration depending on two and five parameters, respectively. The outcomes are compared to those of the full-order CFD model and, in comparison to predictions obtained by comparable proper orthogonal decomposition (POD) based ROMs, an improvement of the results is achieved. Particularly, results featuring a shock are more accurately predicted.

Zusammenfassung

In dieser Arbeit werden neuartige, parametrische Modelle reduzierter Ordnung (engl. *reduced-order models*, ROMs) für stationäre, transonische Anwendungen der Aerodynamik vorgestellt. Dabei basieren die entwickelten ROMs auf den Methoden des Erlernens von Mannigfaltigkeiten (engl. *manifold learning*, ML).

Ziel ist die Herleitung effizienter ROMs, welche unter Ausnutzung der niedrigdimensionalen Mannigfaltigkeit der zugrunde liegenden CFD-Lösungen eine verbesserte Behandlung der Nichtlinearitäten sicherstellen, welche durch variieren von Strömungs- und Geometrieparametern auftreten. Insbesondere soll dies die Vorhersage von Verdichtungsstößen verbessern. Die reduzierte Darstellung der zugrunde liegenden CFD-Daten wird durch die ML-Methode Isomap berechnet, welche für aerodynamische Anwendungen weiterentwickelt wird.

Um ein Modell reduzierter Ordnung zu entwickeln, welches approximierte CFD-Lösungen an bisher nicht abgetasteten Parameterkombinationen vorhersagen kann, wird Isomap mit einem Interpolationmodell gekoppelt. Dadurch wird eine Abhängigkeit des Modells von Veränderungen in den Parametern, wie z.B. dem Anstellwinkel oder der Machzahl, erreicht. Um schließlich hoch-dimensionale Vorhersagen treffen zu können, wird eine lokal-inverse Abbildung von der reduzierten Darstellung in den hoch-dimensionalen CFD-Lösungsraum eingeführt. Dieses Modell wird im späteren Isomap+I genannt. Desweiteren wird die Vorhersage der Isomap+I Methode durch Minimierung des entsprechenden CFD-Flussresiduums verbessert.

Basierend auf der niedrig-dimensionalen Darstellung der Lösungsmannigfaltigkeit, welche durch Isomap berechnet wurde, wird ein Verfahren zum adaptiven Abtasten des Parameterraums entwickelt. Das Ziel dieser Methode ist es, eine gleichmäßig verteilte Datenmannigfaltigkeit zu generieren, um eine bessere Charakterisierung der zugrundeliegenden Mannigfaltigkeit zu erhalten, wodurch die Genauigkeit der Modelle reduzierter Ordnung gesteigert wird.

Die entwickelten ROMs werden, unter anderem in Kombination mit dem Verfahren zum adaptiven Abtasten des Parameterraums, erfolgreich für die Vorhersage von CFD-Lösungen des NACA 64A010 Profils und einer Flugzeug-Konfiguration mit zwei respektive fünf Parametern angewendet. Die Ergebnisse werden mit CFD-Referenzlösungen verglichen, mit denen sie besser übereinstimmen als Vorhersagen von ROMs basierend auf der häufig genutzten POD (engl. *proper orthogonal decomposition*). Insbesondere werden Verdichtungsstöße genauer vorhergesagt.

Preface

As part of this thesis, a refereed publication was written, which appears as [36] in the bibliography. Therein, the ML based ROM Isomap+I was described and applied to 2D and 3D test cases. Since that work is the fundamental building block of this thesis, parts of this publication are reproduced with permission by the publisher. The permission is stated in “Copyright assignment and your rights” of Taylor & Francis:

> “3.2 Retained rights
>
> In assigning Taylor & Francis or the journal proprietor copyright, or granting an exclusive license to publish, you retain: [...]
> - the right to include an article in a thesis or dissertation that is not to be published commercially, provided that acknowledgment to prior publication in the journal is made explicit; [...]”[1]

Text excerpts from [36] are identified by: “*(cited text)*”(Franz et al. [36]).

Work share: The technical supervisor of my PhD thesis, Ralf Zimmermann, proposed the idea of employing ML methods to tackle the shock prediction problem and was involved in the associated theoretical considerations. From this idea, I developed a concrete method and successfully coped with all the theoretical and practical challenges associated with this task. Moreover, I implemented the method including further developments and performed all the numerical experiments. The team leader of DLR’s Aerodynamic Surrogate and Optimization Group, Stefan Görtz, was involved in guiding the strategic direction of the research. The fourth co-author, Niklas Karcher, provided the framework (SMARTy toolbox) for the implementation of the approach.

---

[1] `http://journalauthors.tandf.co.uk/copyright/assignmentAndYourRights.asp` (17. July 2014)

Acknowledgement

In the following, I would like to express my gratitude to many people for supporting me to elaborate this thesis.

First and foremost, I would like to thank Priv.-Doz. Dr. Ralf Zimmermann for supervising my thesis and for making valuable suggestions. I am also grateful to Prof. Dr. Matthias Bollhöfer and Prof. Dr. Volker Schulz for being the co-examiner of this thesis and to Prof. Dr. Dirk Langemann for chairing the PhD committee.

Furthermore, I am very grateful to Prof. Dr. Norbert Kroll, the head of $C^2A^2S^2E$, for providing me the opportunity to pursue a PhD in his department. This thesis has gained from his accurate goal settings and the discussions with him.

I would particularly like to thank my supervisor Priv.-Doz. Dr. Ralf Zimmermann and my team leader Dr. Stefan Görtz for calling my attention to reduced-order modeling. I am deeply grateful to both of them for the valuable guidance as well as the freedom to explore new research paths. Moreover, I am thankful for the many insightful discussions we had, for answering all of my questions and, of course, for proof-reading this thesis.

I am also thankful to Niklas Karcher for providing an excellent framework for an implementation of the developed methods. Furthermore, I gratefully acknowledge the many discussions with him and that he shared his programming knowledge with me to implement an efficient code. Also, I would like to thank Dave Jacobi for providing the Levenberg-Marquardt algorithm.

Special thanks go to my former office mate Stefan Schoenawa and my colleague Christian Lenz for intense discussions and comments from a different point of view. I am also thankful to all colleagues from the $C^2A^2S^2E$ department for the relaxed atmosphere at work and for supporting me, especially to Axel Schwöppe for setting up proper TAU parameter files and to Mohammad Abu-Zurayk for providing a 3D test case and also for coupling the ROM predictions to the structural sizing process.

I would like to thank my friends, especially Marcel Wallraff, Jan Grippenkoven and Niklas Karcher, for creating the necessary counterbalance to my work. My family deserves special thanks for their continuous support and believe in me. Finally I would like to thank Betti for not only creating the necessary counterbalance to my work, but also for her patience and understanding.

Contents

Acronyms

API	application programming interface
BLUP	best linear unbiased predictor
CFD	computational fluid dynamics
CVT	centroidal Voronoi tessellation
DEIM	discrete empirical interpolation method
DLR	Deutsches Zentrum für Luft- und Raumfahrt e. V.
DNS	direct numerical simulation
DoE	design of experiment
DoF	degree of freedom
DR	dimensionality reduction
EIM	empirical interpolation method
EN-Isomap	edge number-based Isomap
EVD	eigenvalue decomposition
FLOP	floating-point operation
FSDM	FlowSimulator DataManager
GNAT	Gauss-Newton with approximated tensors
HOSVD	higher order singular value decomposition
HYE	hybrid error
LDEIM	localized discrete empirical interpolation method
LES	large-eddy simulation
LHS	Latin hypercube sampling
LLC	locally linear coordination
LLE	locally linear embedding

LU-SGS	lower-upper symmetric Gauss-Seidel
MDE	maximum distance error
MDO	multidisciplinary optimization
MDS	multidimensional scaling
ML	manifold learning
MPE	missing point estimation
MSE	mean squared error
ODE	ordinary differential equation
PCA	principal component analysis
PDE	partial differential equation
POD	proper orthogonal decomposition
RANS	Reynolds-averaged Navier-Stokes
RB	reduced basis
RBF	radial basis functions
ROB	reduced-order basis
ROM	reduced-order model
SA	Spalart-Allmaras
SVD	singular value decomposition
TAU	turbulent adaptive unstructured
TPS	thin plate spline

List of Notations

General notations

$\boldsymbol{u}, \boldsymbol{U}$	vector in $\mathbb{R}^n$
u_i, U_i	ith component of a vector $\boldsymbol{u}, \boldsymbol{U} \in \mathbb{R}^n$
A	matrix in $\mathbb{R}^{n \times m}$
$\boldsymbol{A}^i$	ith column of a matrix $A \in \mathbb{R}^{n \times m}$
$A_{ij} = A_i^j$	entry in the i-the row and the j-the column of the matrix $A \in \mathbb{R}^{n \times m}$

Specific notations

$\boldsymbol{a}$	vector of the POD coefficients or of the back-mapping weights
$\mathcal{C}^r$	set of continuous functions that have continuous first r derivatives
C_L	lift coefficient
C_p	pressure coefficient
d	dimension of the parameter space
E	set of edges of a graph G
$\mathtt{E}$	total energy per unit mass
F^c	convective fluxes
F^v	viscous fluxes
G	undirected weighted neighborhood graph
I	identity matrix
$\mathcal{I}$	index set of the nearest neighbors
Ma	Mach number

n_g	grid size; element of $\mathbb{N}$
n_v	number of variables; element of $\mathbb{N}$
$n = n_g n_v$	dimension of state space; element of $\mathbb{N}$
$\boldsymbol{n}$	outward-facing unit normal vector
p	pressure
$\mathcal{P}$	parameter space; subset of $\mathbb{R}^d$
P	discrete subset of $\mathcal{P}$
$\boldsymbol{p}$	parameter combination; element of $\mathcal{P}$
$\boldsymbol{q}$	heat flux vector
$\mathbb{R}$	set of all real numbers
$\mathbb{R}^n$	n-dimensional Euclidean space
Re	Reynolds number
$\mathbf{res}(\cdot)$	discretized flux residual; $\mathbb{R}^n \to \mathbb{R}^n$
$s(\cdot)$	stress function to measure the quality of the embedding with respect to the neighborhood size
$\boldsymbol{U}^j$	jth proper orthogonal decomposition mode; element of $\mathbb{R}^n$
V	set of vertices of a graph G
$\boldsymbol{v}$	velocity vector
$\boldsymbol{W}$	flow solution state vector (snapshot); element of $\mathbb{R}^n$
W	snapshot matrix; element of $\mathbb{R}^{n\times m}$
$\mathcal{W}$	flow solution manifold; submanifold of $\mathbb{R}^n$
$\boldsymbol{y}$	low-dimensional embedding vector; element of $\mathbb{R}^d$
Y	matrix containing the low-dimensional embedding vectors; subset of $\mathbb{R}^d$

Greek symbols

α	angle of attack
δ_{ij}	Kronecker delta
ρ	density
σ	singular value

τ	viscous stress tensor
Ω	diagonal matrix of grid cell volumes; element of $\mathbb{R}^{n\times n}$
$\varOmega$	open, bounded, and connected computational domain
$\partial\varOmega$	boundary of domain $\varOmega$
$\omega(\cdot)$	indicator function (5.3)

Mathematical operators

$\langle \boldsymbol{u},\boldsymbol{v}\rangle$	$= \sum_{i=1}^{n} u_i v_i$	Euclidean scalar product of two vectors $\boldsymbol{u},\boldsymbol{v}\in\mathbb{R}^n$
$\parallel \boldsymbol{u} \parallel_2$	$= \sqrt{\langle \boldsymbol{u},\boldsymbol{u}\rangle}$	Euclidean norm (2-norm) of a vector $\boldsymbol{u}\in\mathbb{R}^n$
$\langle f,g\rangle_{L_2}$	$= \int_{\varOmega} f g \, d\varOmega$	L_2 scalar product of real-valued functions f and g
$\parallel f \parallel_{L_2}$	$= \sqrt{\langle f,f\rangle_{L_2}}$	L_2 norm of a real-valued function f
$\langle A,B\rangle_F$	$= \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij}$	Frobenius scalar product for matrices $A,B\in\mathbb{R}^{n\times m}$
$\parallel A \parallel_F$	$= \sqrt{\langle A,A\rangle_F}$	Frobenius norm for a matrix $A\in\mathbb{R}^{n\times m}$
∇	$= (\frac{\partial}{\partial x_1},\dots,\frac{\partial}{\partial x_n})^T$	vector of partial differential operators for each space dimension (nabla operator)
$\nabla u(\boldsymbol{x})$	$= (\frac{\partial}{\partial x_1},\dots,\frac{\partial}{\partial x_n})^T u(\boldsymbol{x})$	gradient of u
$\mathrm{conv}(Y)$		convex hull of a finite point set $Y=\{y^1,\dots,y^m\}$
$\mathrm{span}(Y)$		span of a finite point set $Y=\{y^1,\dots,y^m\}$

CHAPTER 1

Introduction

A common situation in many industrial applications is the need to gain aerodynamic data of an underlying model for a vast amount of different design parameters. Since setting up and evaluating wind-tunnel campaigns for all parameter configurations is quite expensive in terms of time and costs, a first step in reducing the time and costs in the field of aerodynamics was replacing experiments by computational fluid dynamics (CFD) calculations. In this case the underlying models are described by partial differential equations (PDEs), e.g., the Navier-Stokes or Euler equations, which are numerically solved up to a user-defined accuracy. However, if the number of parameter configurations is large and a fully nonlinear behavior is involved, it is still time expensive to obtain accurate numerical solutions. In aerodynamic engineering applications such as design, optimization, or aero-loads prediction over the entire flight envelope, a large number of solutions to the compressible Navier-Stokes equations at varying flow conditions and for changing geometries are required. Since industrial development cycles most often do not permit to compute all the required aerodynamic data via high-fidelity CFD, fast, yet sufficiently accurate reduced-order models (ROMs) are sought after.

1.1. Literature review

A simple and obvious idea is to interpolate solutions between a limited number of previously computed solutions, called snapshots herein, at well-selected parameter combinations. However, the accuracy of the interpolation model degrades when the distance between the sampled parameter combinations is too large or when the problem depends on too many parameters [54].

An alternative method to plain interpolation, called POD+I [56, 21, 34], combines the application of interpolation and proper orthogonal decomposition (POD) [45, Section 3], also referred to as principal component analysis (PCA) [80]. POD is currently considered state-of-the-art for the order reduction of large-scale systems and is mathematically related to the singular value decomposition (SVD). It yields a set of basis functions, called POD modes, which describe the dominant behavior of the dynamics and the most relevant coherent structures of a given problem. The POD is also used within a Galerkin projection framework (POD-Galerkin) [55], or it is combined with a CFD flux residual minimization scheme [53, 94]. "Although POD is applicable to nonlinear problems, its main drawback is the underlying assumption that the flow solutions lie in (or at least close to) a low-dimensional, *linear subspace* of the high dimensional space spanned by the full-order CFD flow solutions. Therefore, highly nonlinear features are often insufficiently reproduced. In particular, accurately predicting shocks in the transonic flow regime poses a severe challenge for POD-based methods because such flows result in sharp, discontinuous changes in flow variables such as pressure, temperature, density, and velocity across the shock."(Franz et al. [36])

A nonlinear extension of POD, called kernel POD, was applied to a 2D airfoil by Mifsud [59]. Using a Gaussian kernel, weak nonlinearities were modeled adequately, but, however, highly nonlinear features like the moving shock wave on the surface of the airfoil was not well approximated for that test case.

"Projection-based ROM techniques like POD-Galerkin, which reformulate the system of partial differential equations in its continuum or discretized version by projection operators, can be regarded as intrusive methods [34]. State-of-the-art methods belonging to this class of ROMs are the Gauss-Newton with approximated tensors (GNAT) method [22] and discrete empirical interpolation method (DEIM) [25]. The GNAT method uses the Gauss-Newton method with approximated tensors in a projection based framework associated with residual minimization, whereas DEIM is based on the standard POD-Galerkin approach, but also reduces the complexity of evaluating the nonlinear term of the governing equations by using an additional POD basis for this purpose.

[...] Nonintrusive methods like POD+I where the explicit projection phase is avoided, making the models independent of the full-order equations, may thus be constructed given any black-box solver. Moreover, these ROMs are outstandingly cheap to evaluate."(Franz et al. [36])

An higher order singular value decomposition (HOSVD) based ansatz introducing a special treatment of shock waves for 2D airfoils was developed by Lorente et al. [54]. The main idea of this treatment is as follows: 1) Localization of the shock position, identification of its structure and disassembling of the shock region and the smooth region 2) Application of HOSVD to each region 3) Interpolation and assembling of the separated elements. This idea was extended by Alonso et al. [3] and adapted to POD based methods including residual minimization. Alonso conluded from the results of his method applied to a specific 2D airfoil that the method is “reasonably good, except for parameter values in the shock wave existence/nonexistence boundary, where more snapshots would be needed”. An alternative treatment based on perturbation methods was developed in the 80’s by Stahara et al. [83]. The procedure defines a unit perturbation between a “base” and a “configuration solution” which have a nominal change in some geometric or flow parameter. Then it predicts solutions for new parameters by adding the unit perturbation to the base solution, weighted by an incremental change of the varied parameter. The method was successfully applied to settings depending on a variety of flow and geometric parameters like angle of attack, Mach number and thickness-ratio, but, due to the nature of this method, in each setting there was only one varying parameter, all others were fixed. Therefore, it is not competitively viable for today’s challenges anymore.

An alternative application of the POD was recently developed by Amsallem et al. [4], which might also be convenient to tackle the typical problems of the POD for shock dominated CFD snapshots. For unsteady problems, instead of approximating the ROM solution in a fixed lower-dimensional POD subspace of global POD modes, this method approximates the solution in a lower-dimensional subspace generated by local POD modes, which leads to better-shaped local modes. Their results show that in general the local framework ends in a more accurate and computationally more efficient nonlinear ROM than its counterpart, the standard ROM based on global POD modes.

A similar local method for dynamical systems, called Trajectory Piecewise-Linear (TPWL) reduced order systems, was published by Rewienski [67]. It relies on a quasi-piecewise-linear approximation of nonlinearities, where the linearizations points are selected from a state-space 1D trajectory. The TPWL approach was successfully applied to several nonlinear systems, especially for electronic circuit simulations [68, 69, 13]. Gratton and Willcox adapted TPWL to the 2D Euler equations to derive a nonlinear CFD ROM [41]. In their approach, they use POD bases for the local linearizations around the selected points and then build a global nonlinear model by a weighted sum of the linearized models. The major drawback of this method for many query applications with more than one parameter is the trajectory. Even when going to a problem with two parameters the trajectory becomes a surface, which makes it more complicated to choose the linearizations points. Additionally, there are still many nonlinear functions that cannot be well approximated by low-degree piecewise polynomials.

Alternative ideas to improve predictions by restriction to local snapshot information were pursued by Eftang et al. [31] and Peherstorfer et al. [63]. In the former, a hierachical splitting of the parameter domain into smaller subdomains based on proximity is obtained. Afterwards, local standard reduced basis (RB) approximations spaces are constructed on each subdomain. Furthermore, Greedy sampling procedures and a posteriori error estimation are exploited. In contrast, the localized discrete empirical interpolation method (LDEIM) by Peherstorfer et al. "computes several local subspaces, each tailored to a particular region of characteristic system behavior. Then, depending on the current state of the system, LDEIM selects an appropriate local subspace for the approximation of the nonlinear term."[63] LDEIM was demonstrated for a reacting flow example of an H_2-Air flame, where its accuracy was about two orders of magnitude better than standard DEIM.

1.2. Objective of this thesis

The highly nonlinear dependency of the CFD solutions on the parameters, e. g., the angle of attack, the Mach number or various geometry parameters, makes the derivation of a robust and accurate ROM quite difficult. Especially ROMs that are based on methods with linear assumptions like POD are restricted to parameter spaces with slight variation of the parameters or the underlying sampling has to be very dense.

The objective of this thesis is to construct new robust and accurate ROMs applicable in the transonic flow regime, i. e., featuring an improved shock prediction behavior to obtain more accurate steady-state solutions for an acceptably large parameter space. Furthermore, the predictions should be obtained in a sufficiently small amount of time, which is relevant to (near-)real-time applications. "To this end, the linear order reduction technique of POD is replaced by the nonlinear manifold learning (ML) method Isomap [85]. ML, or, more generally, nonlinear dimensionality reduction (DR), is widely used in the field of data and image processing. In contrast to POD, the basic assumption here is that the full-order data lies on a *nonlinear manifold* of low dimension.

The transfer of these nonlinear dimensionality reduction methods to fluid flow problems, however, poses several challenges:
(1) In the context of data and image processing, only the mapping from the high-dimensional snapshot space to a low-dimensional representation is of interest, while ROMs of the Navier-Stokes equations are required to deliver approximate flow solutions of the same type and dimension as the full-order CFD solver. Hence, the back-mapping (low-dimensional space to high-dimensional space) becomes essential.
(2) For data analysis applications, there is usually a vast amount of full-order input data. In contrast, when constructing CFD-based aerodynamic ROMs, the goal is to compute as few full-order solutions as possible due to the high cost of computing a

single CFD solution.
(3) The main application of the established dimensionality reduction methods is data *compression* or *visualization*. For the aerodynamic applications envisioned in this work, however, the reduced-order *prediction* of the aerodynamic loads at an untried parameter combination is to be computed.

In the work at hand, [all of the above challenges are tackled]. To this end, the Isomap algorithm is combined with a spatial interpolation method. As indicated above, the back-mapping from the reduced space back to the full-order space is not as straight-forward as for the POD method. The idea for this mapping is based on locally linear embedding (LLE) [70] and is also outlined in [73, Section 6.1]. In order to cope with the relatively small number of full-order CFD solutions used to construct the ROM, a new penalty term is introduced, which decreases the influence of snapshots far away from the prediction point on the ROM approximation. It is important to emphasize that the distance between snapshots is measured with respect to the metric of the manifold, which is approximated via Isomap. Altogether, in this way an Isomap-based reduced-order model (Isomap+I) similar to POD+I is obtained, which can be used to predict approximate flow solutions at untried parameter combinations. Since the back-mapping only uses an affine combination of the nearest neighbors on the estimated manifold to compute a prediction, it is expected that the Isomap+I ROM approximates local features more accurately than POD-based models."(Franz et al. [36])

A CFD-enhanced back-mapping based on CFD flux residual minimization is also developed later on. It compromises the extra costs of solving an optimization problem by adding additional physics to the prediction. A starting solution for the optimization process is given by the Isomap+I prediction. As before, this back-mapping is only based on a few nearest neighbors.

However, the quality of the predictions of both methods depends on the set of metaparameters required for the Isomap-based ROMs and on the sampled parameters since Isomap relies on a good discription of the underlying solution manifold. Thereto, an adaptive sampling strategy is developed to generate a snapshot set that approximately constitutes a homogeneously distributed sampling of the manifold to prevent a lack of information regarding the geometry of the manifold. Furthermore, to provide a user-friendly ROM, several methods which automatically determine a proper set of the essential metaparameters of the Isomap based ROMs are developed.

1.3. Thesis outline

This thesis is structured as follows. In Chapter 2, a brief introduction to the governing PDEs is given. Furthermore, since a set of solutions to the governing equations is necessary to construct the ROMs developed in this thesis, the applied CFD code is described.

Chapter 3 provides an overview of the reduced-order modeling methods currently considered state-of-the-art. First, the DR method POD is developed in Section 3.1. Afterwards, in Sections 3.2, 3.3 and 3.4, different techniques that exploit the POD to construct ROMs are outlined.

The novel ML-based ROMs are introduced in Chapter 4. Since the employed ML method Isomap goes back to the theory of manifolds and Riemannian geometry, a brief overview of this theory is outlined in Section 4.1. The methodology of ML along with an overview of various ML methods is presented in Section 4.2. In Section 4.3 and Section 4.4, the essential ingredients of the novel ROMs are detailed. More precisely, Section 4.3 introduces the employed ML method Isomap and in Section 4.4 the back-mapping from the low-dimensional space to the full-order CFD space is derived. Two different kinds of ROMs exploiting the previously described essential ingredients are formulated in Section 4.5 and Section 4.6. The last Section 4.7 of Chapter 4 gives an idea of additional possible applications of the previous methods.

Most ROMs rely on a well-sampled set of snapshots to yield accurate predictions. The same holds for the novel methods introduced in this thesis and thus a novel adaptive sampling strategy exploiting the characteristics of the solution manifold discovered by Isomap is derived in Chapter 5. The method itself is developed in Section 5.1 and a proof of concept is given in Section 5.2.

Computational issues regarding the derived ROMs are discussed in Chapter 6. In Section 6.1, the framework of choice for the implementation of the methods is described and the implementation details are outlined. The computational complexity of the novel ROMs is analyzed in Section 6.2. At last, notes on the parallelization of the ROMs are given in Section 6.3.

The novel techniques introduced in this thesis are applied to two different test cases in Chapter 7. The first test case, a 2D test case, is described in the introduction of Section 7.1. Both novel ROMs are applied to a randomly sampled set of snapshots in order to predict solutions at untried parameter combinations. The results are compared to reference CFD solutions and to predictions obtained by POD-based ROMs in Section 7.1.1 and Section 7.1.2. In Section 7.1.3, the adaptive sampling strategy is applied to obtain a second set of snapshots and is compared to various other sampling strategies. The 3D test case is outlined in Section 7.2. Both novel ROMs are also applied to the 3D test case and compared to reference CFD solutions and to predictions obtained by POD-based ROMs in Section 7.2.1 and Section 7.2.2. The underlying set of snapshot is generated by the developed adaptive sampling strategy.

The final conclusions are given in Chapter 8.

Additional information is provided in the appendix. In Appendix A, the derivations of radial basis functions (RBF) models is outlined. The definition of the Delaunay triangulation is given in Appendix B.

CHAPTER 2

Computational fluid dynamics

The construction of a ROM in aerodynamics usually requires the generation of a set of aerodynamic flow field data. Each flow field *snapshot* can be obtained either by experiments or by numerical simulations. In the work at hand, the snapshots are solely calculated by solving the underlying PDEs, i. e., the Euler, Navier-Stokes or Reynolds-averaged Navier-Stokes (RANS) equations, which are briefly discussed in Section 2.1. A semi-discretized formulation comprising the different types of governing equations in one system of ordinary differential equations (ODEs) is also given in the first section. The governing equations are solved by the DLR flow solver TAU [75] (Section 2.2) to obtain the required snapshots.

2.1. Governing equations

The derivation of the Navier-Stokes equations is based on the basic physical principles that rule the motion of a fluid: the conservation of mass, momentum and energy. Assembling these conservation laws to one system, the *Navier-Stokes equations* in integral form [19, Section 2.4] read in absence of source terms as

$$\frac{\partial}{\partial t}\int_{\Omega} \boldsymbol{W}\, d\Omega + \oint_{\partial\Omega} \langle F^c(\boldsymbol{W}) - F^v(\boldsymbol{W}, \nabla\boldsymbol{W}), \boldsymbol{n}\rangle \; d(\partial\Omega) = \boldsymbol{0}, \tag{2.1}$$

or, by applying Gauss' theorem, in differential form [19, Section A.1] as

$$\frac{\partial}{\partial t}\boldsymbol{W} + \langle \nabla, F^c(\boldsymbol{W}) - F^v(\boldsymbol{W}, \nabla\boldsymbol{W})\rangle = \boldsymbol{0}, \tag{2.2}$$

where Ω is the domain, $\partial\Omega$ is its boundary and the scalar product in (2.1) is conducted with the unit normal vector $\boldsymbol{n}$. The vector

$$\boldsymbol{W} = \begin{pmatrix} \rho \\ \rho v_1 \\ \rho v_2 \\ \rho v_3 \\ \rho \mathtt{E} \end{pmatrix} \tag{2.3}$$

is the vector of *conservative variables* in three spatial dimensions, where ρ is the density, $\mathtt{E}$ the total energy per unit mass and v_1, v_2, v_3 are the components of the velocity vector $\boldsymbol{v} = (v_1, v_2, v_3)^T$. The *convective flux* $F^c(\boldsymbol{W}) = (\boldsymbol{f}_1^c(\boldsymbol{W}), \boldsymbol{f}_2^c(\boldsymbol{W}), \boldsymbol{f}_3^c(\boldsymbol{W}))$ and the *viscous flux* $F^v(\boldsymbol{W}, \nabla\boldsymbol{W}) = (\boldsymbol{f}_1^v(\boldsymbol{W}, \nabla\boldsymbol{W}), \boldsymbol{f}_2^v(\boldsymbol{W}, \nabla\boldsymbol{W}), \boldsymbol{f}_3^v(\boldsymbol{W}, \nabla\boldsymbol{W}))$ are given by

$$\boldsymbol{f}_i^c(\boldsymbol{W}) = \begin{pmatrix} \rho v_i \\ \rho v_1 v_i + \delta_{1i} p \\ \rho v_2 v_i + \delta_{2i} p \\ \rho v_3 v_i + \delta_{3i} p \\ (\rho \mathtt{E} + p) v_i \end{pmatrix}, \quad i = 1, 2, 3, \tag{2.4}$$

and

$$\boldsymbol{f}_i^v(\boldsymbol{W}, \nabla\boldsymbol{W}) = \begin{pmatrix} 0 \\ \tau_{1i} \\ \tau_{2i} \\ \tau_{3i} \\ \sum_{j=1}^{3} \tau_{ij} v_j - q_i \end{pmatrix}, \quad i = 1, 2, 3, \tag{2.5}$$

where δ_{ij} is the Kronecker symbol and p, $\tau = (\tau_{ij})_{i,j=1,\ldots,3}$ and $\boldsymbol{q} = (q_1, q_2, q_3)^T$ denote the pressure, the viscous stress tensor and the heat flux, respectively. Assuming that the working fluid behaves like a calorically *perfect gas*, the associated formula for the pressure [19, Section 2.4.1],

$$p = (\gamma - 1)\rho\Big(\mathtt{E} - \frac{v_1^2 + v_2^2 + v_3^2}{2}\Big), \tag{2.6}$$

with $\gamma = 1.4$ for air, closes the system of equations.

If the viscous forces in Equation 2.1 are neglected, meaning that the viscous flux F^v is omitted, then the equations are called the *Euler equations* [19, Section 2.4.3] and read

$$\frac{\partial}{\partial t} \int_\Omega \boldsymbol{W} \, d\Omega + \oint_{\partial\Omega} \langle F^c(\boldsymbol{W}), \boldsymbol{n} \rangle \; d(\partial\Omega) = \boldsymbol{0}. \tag{2.7}$$

A direct simulation of small-scale fluctuations of a turbulent flow governed by (2.1), called direct numerical simulation (DNS), is applicable only to relative simple flow problems at low Reynolds numbers [19, Chapter 7]. To treat flow problems at higher Reynolds numbers, the control volumes have to be kept very small which leads to fine grids and hence to enormous computational costs which prevents a more widespread application of DNS. Therefore, an approximate treatment of turbulent flows is necessary.

One approach is to split each flow variable in a mean and a fluctuation part, which is called *Reynolds averaging.* Inserting this approach into the Navier-Stokes equations (2.1), the RANS equations [19, Section 7.1.3], an underdetermined set of PDEs, are obtained. In order to close the system, additional relations have to be established by a *turbulence model*. There are five principal classes of turbulence models: algebraic, one-equation, multiple-equation, *second-order* closures and large-eddy simulation (LES), whereby the first three types belong to the class of *first-order* closures. A detailed overview is given in [19, Chapter 7].

In this work, the governing equations of fluid dynamics, more precisely the Euler (2.7), Navier-Stokes (2.1) and RANS [19, Section 7.1.3] equations, are considered in a generalized notation. To this end, the PDEs are "[...] spatially discretized on a grid of size n_g for some aerodynamic configuration. Let n_v be the corresponding number of primitive mean-flow variables plus the number of primitive variables associated with the turbulence model, if present. The *primitive mean-flow variables* are the density, ρ, the velocity components in all spatial directions, v_x, v_y, v_z, and the total energy, E. The number of primitive turbulence variables depends on the chosen turbulence model. Let $n = n_g n_v$ denote the total length of the discretized flow solution vectors."(Franz et al. [36]) Given a finite volume scheme, the corresponding system of semi-discrete ODEs can be written as

$$\frac{d}{dt}\boldsymbol{W}(t;\boldsymbol{p}) + \Omega^{-1}\mathbf{res}(\boldsymbol{W}(t;\boldsymbol{p})) = \mathbf{0} \in \mathbb{R}^n, \tag{2.8}$$

where $t \in \mathbb{R}^+$ denotes the time, $p \in \mathbb{R}^d$ is the vector of input parameters, $\boldsymbol{W}(t;\boldsymbol{p}) \in \mathbb{R}^n$ is the state vector of primitive variables and $\mathbf{res}(\boldsymbol{W}(t;\boldsymbol{p})) \in \mathbb{R}^n$ is the vector of flux residuals corresponding to the state solution $\boldsymbol{W}(t;\boldsymbol{p})$. The diagonal matrix $\Omega \in \mathbb{R}^{n\times n}$ with n_v sub-blocks, each containing the cell volumes $(vol_1, \dots, vol_{n_g})$ of the corresponding computational grid on the diagonal, stems from the spatial discretization of the flow problem and induces the discrete L_2-metric.

Denoting by $W_{k,i}$ the value of flow variable k corresponding to grid cell i and by $\text{res}_{k,i}(\boldsymbol{W})$ the flux residual of flow variable k in grid cell i, the system (2.8) may be written in cell-wise form as

$$\forall k = 1, \ldots, n_v \;\; \forall i = 1, \ldots, n_g : \;\; \frac{d}{dt} W_{k,i} + \frac{1}{vol_i} \mathrm{res}_{k,i}(\boldsymbol{W}) = \mathbf{0}. \tag{2.9}$$

The steady-state is achieved if the time derivative drops out in equations (2.8) or (2.9), or, equivalently, if the CFD flux residual vanishes:

$$\mathbf{0} = \Omega^{-1} \mathbf{res}(\boldsymbol{W}(\boldsymbol{p})) \in \mathbb{R}^n. \tag{2.10}$$

When computing approximations to (2.10) restricted to a subspace of lower dimension, the weighting Ω^{-1} is significant (see Section 3.4).

2.2. Solution method

The DLR turbulent adaptive unstructured (TAU) code [75, 51] is utilized to solve the Euler equations (2.7) or the RANS equations endowed with a turbulence model by a finite volume approach, employing hybrid unstructured grids. However, the system of equations has to be completed by boundary conditions like wall, farfield or symmetry boundary before, where values of certain physical quantities have to be specified. A survey of various boundary conditions is given in [19, Chapter 8]. The underlying equations are then discretized by the *method of lines*, i. e., a separate discretization in time and space. The spatial discretization is accomplished by an edge-based dual-cell approach, i. e., a vertex-centred scheme, where inviscid terms are computed by a variety of upwind schemes using linear reconstruction for second-order accuracy or a second-order central scheme. The latter is also used for the viscous terms. The resulting system of ODEs is then integrated in time by an explicit Runge-Kutta or an implicit lower-upper symmetric Gauss-Seidel (LU-SGS) method with convergence acceleration by a multi-grid algorithm to obtain a steady-state solution. A detailed description of the functionality of the DLR TAU code and its modules is given in [75].

CHAPTER 3

Reduced-order modeling via POD - state of the art

In this chapter, the currently considered state-of-the-art methods for deriving ROMs in the field of CFD are described. The underlying process is the POD [86, 74], also known as Karhunen-Loève decomposition or PCA [80], which is a powerful method for data analysis. It was introduced independently by numerous people at different times including Kosambi [49], Loève and Karhunen [47] and was successfully applied to various disciplines other than fluid dynamics like structural dynamics [1], data compression [6] and forecasting in meteorology [11].

POD will be briefly discussed in Section 3.1. Afterwards, three different techniques to derive a POD based ROM are described. A common technique is to project the underlying spatially discretized PDEs onto the POD subspace to obtain a system of ODEs (Section 3.2). An interpolation based method will be presented in Section 3.3. Finally, the POD will be integrated into an optimization problem, which minimizes the residual of the underlying equations.

3.1. Proper orthogonal decomposition

> "The attractiveness of the POD lies in the fact that it is a linear procedure. The mathematical theory behind it is the spectral theory of compact, self-adjoint operators. This robustness makes it a safe haven in the intimidating world of nonlinearity; although this may not do the physical violence of linearization methods, the linear nature of the POD is the source of its limitations [...]."
>
> *by Berkooz, Holmes, and Lumley [16, Section 1.3]*

Nowadays the derivation of ROMs is frequently done by using the POD, which is a powerful method for data analysis for multivariate and nonlinear phenomena. Its objective is to determine low-dimensional approximative descriptions of high-dimensional processes, which contain the most dominant features and trends of the underlying data, generated by numerical simulations or experimental approaches. Motivated by the idea of approximating a function by a finite sum, i.e.

$$f(\boldsymbol{x}, t; \boldsymbol{p}) \simeq \sum_{k=1}^{M} a_k(t; \boldsymbol{p}) \phi_k(\boldsymbol{x}), \tag{3.1}$$

the POD method generates a sequence of orthogonal basis functions ϕ_i, which span an optimal linear subspace for the corresponding solution space.

Here, a brief introduction to the POD in the context of finite dimensional vector spaces is given, following the view of Chatterjee [24]. For a more comprehensive description, see Holmes et al. [45] or Tropea et al. [86].

Let $W = (\boldsymbol{W}^1, \dots, \boldsymbol{W}^m) \in \mathbb{R}^{n\times m}$ be a data matrix consisting of m measurements of dimension n, called *snapshots*, obtained by numerical simulations or experimental approaches. By computing the SVD [39, Section 2.4] of the matrix W, we obtain

$$W = U\Sigma V^T, \tag{3.2}$$

where $U \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{m\times m}$ are orthogonal matrices, $\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_p, 0, \dots, 0) \in \mathbb{R}^{n\times m}$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p \geq 0$ and $p = \min(n, m)$. The nonnegative numbers σ_i are called the *singular values* of W.

Remark. The squared singular values $\sigma_1^2, \dots, \sigma_p^2$ are the non-zero eigenvalues of W^TW and WW^T with corresponding eigenvectors $\boldsymbol{V}^1, \dots, \boldsymbol{V}^p$ and $\boldsymbol{U}^1, \dots, \boldsymbol{U}^p$, respectively.

3.1 Corollary. *W has r strictly positive singular values, i.e. $\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_p = 0 \Leftrightarrow \operatorname{rank}(W) = r$.*

For $n \geq m \geq r := \operatorname{rank}(W)$, the representation of each snapshot $\boldsymbol{W}^i$ is given by the finite sum

$$\boldsymbol{W}^i = U\Sigma(V^T)^i = U\Sigma(V_i^1, \dots, V_i^m)^T = \sum_{j=1}^{m} (\sigma_j V_i^j)\boldsymbol{U}^j = \sum_{j=1}^{r} (\sigma_j V_i^j)\boldsymbol{U}^j, \tag{3.3}$$

which is the discrete form of equation (3.1). The orthogonal basis $\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^r\}$ given by the first r columns of U features the property

$$\operatorname{span}\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^r\} = \operatorname{span}\{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\}, \tag{3.4}$$

where the set $\{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\}$ denotes the columns of W. The $\boldsymbol{U}^i$ are called the *POD modes*, the $a_j^i := \sigma_j V_i^j = \langle \boldsymbol{W}^i, \boldsymbol{U}^j \rangle$, $j = 1, \dots, r$, are the *POD coefficients* of $\boldsymbol{W}^i$ and $\operatorname{span}\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^r\}$ is the *POD subspace*.

Remark. Often it is best to find an affine POD subspace instead of a linear one [64]. For this purpose the mean of all snapshots is subtracted from each column of W:

$$\hat{\boldsymbol{W}}^i = \boldsymbol{W}^i - \tfrac{1}{m}\sum_{j=1}^{m} \boldsymbol{W}^j, \quad i = 1, \ldots, m. \tag{3.5}$$

Hence, when considering unsteady flows, the POD captures only the small fluctuation around the mean flow. Since the centered snapshots are linearly dependent, only $r-1$ POD modes and the mean flow are needed to reconstruct the snapshots:

$$\boldsymbol{W}^i = \tfrac{1}{m}\sum_{j=1}^{m} \boldsymbol{W}^j + \sum_{j=1}^{r-1} (\hat{\sigma}_j \hat{V}_i^j) \hat{\boldsymbol{U}}^j, \tag{3.6}$$

where $\hat{W} = (\hat{\boldsymbol{W}}^1, \ldots, \hat{\boldsymbol{W}}^m) = \hat{U}\hat{\Sigma}\hat{V}^T$.

Lower-rank Approximation. A lower-rank approximation to W of order $k < r$ is obtained by setting $\sigma_{k+1} = \sigma_{k+2} = \ldots = \sigma_r = 0$ (see Corollary 3.1), i.e.

$$W_k = U\Sigma_k V^T \tag{3.7}$$

with $\Sigma_k = \operatorname{diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)$, whereby W_k denotes the lower-rank approximation to W. This approximation is optimal in the following sense:

3.2 Theorem (The Eckhart-Young Theorem [39, Section 2.4])**.** *Let $W, X \in \mathbb{R}^{n\times m}$. If $k < r = \operatorname{rank}(W)$ and $W_k = U\Sigma_k V^T = \sum_{i=1}^{k} \sigma_i \boldsymbol{U}^i (\boldsymbol{V}^i)^T$, then*

$$\min_{\operatorname{rank}(X)=k} \|W - X\|_2^2 = \|W - W_k\|_2^2 = \sigma_{k+1}^2, \tag{3.8}$$

$$\min_{\operatorname{rank}(X)=k} \|W - X\|_F^2 = \|W - W_k\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{3.9}$$

Therefore, the POD basis $\{\boldsymbol{U}^1, \ldots, \boldsymbol{U}^k\}$ leads to the best rank-k approximation $W_k = U\Sigma_k V^T$ to W with respect to either of the norms $\|\cdot\|_2$ and $\|\cdot\|_F$. Moreover, the associated orthogonal projection minimizes the distance to the original data set in the following sense:

$$U = \underset{\substack{Q\in\mathbb{R}^{n\times m} \\ Q^TQ=I}}{\arg\min} \|W - QQ^TW\|_F. \tag{3.10}$$

A criterion, to find a good trade-off between the dimension k and the quality of the approximation, can be deduced from Theorem 3.2. It states, that the error of a rank-

k approximation is $\sum_{i=k+1}^{r} \sigma_i^2$ (eq. (3.9)), which are the singular values belonging to $\boldsymbol{U}^{k+1}, \dots, \boldsymbol{U}^r$. Therefore, the *relative information content of the jth mode* is defined as the ration $r_j = \frac{\sigma_j^2}{\sum_{i=1}^{r} \sigma_i^2}$ and the *relative information content* as

$$RIC(k) = \sum_{i=1}^{k} r_i = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}. \tag{3.11}$$

In practical applications, the dimension k of the POD subspace is chosen such that $RIC(k) \geq 1 - \epsilon$, $\epsilon \in (0, 1]$.

POD in practice. In CFD problems, the dimension of the solution space is much larger than the number of snapshots ($n \gg m$). Thus, performing a full SVD is too expensive, but instead, considering the remark on page 14, the eigenvalue decomposition (EVD) of $W^T W \in \mathbb{R}^{m \times m}$ is calculated to obtain the eigenvalues $\lambda_i = \sigma_i^2$ and eigenvectors V^i. The POD modes are then given by

$$U^i = \tfrac{1}{\sigma_i} W \boldsymbol{V}^i = \tfrac{1}{\sqrt{\lambda_i}} W \boldsymbol{V}^i. \tag{3.12}$$

This method is called the *method of snapshots* [81]. Note that SVD is numerical more advantageous than $W^T W$, nevertheless $W^T W$ is easier to parallelize than the SVD.

Without loss of generality it is assumed in the next sections, that the m measurements/snapshots $W^1, \dots, W^m$ are linearly independent.

3.2. Galerkin projection onto POD subspace

Consider the spatially discretized equations (2.8) and a set of m linearly independent snapshots $\{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\}$ obtained by solving the underlying governing equations at different parameter combinations $(t^1; \boldsymbol{p}^1), \dots, (t^m; \boldsymbol{p}^m) \in \mathcal{T} \times \mathcal{P}$, i. e., $\boldsymbol{W}(t^i; \boldsymbol{p}^i) = \boldsymbol{W}^i$ for $i = 1, \dots, m$, where $\mathcal{T} \subset \mathbb{R}^+$ and $\mathcal{P} \subset \mathbb{R}^d$ is a predefined parameter space. The POD modes $\boldsymbol{U}^1, \dots, \boldsymbol{U}^m$ of the corresponding matrix $W := (\boldsymbol{W}^1, \dots, \boldsymbol{W}^m)$ with $\operatorname{rank}(W) = m$ are calculated as stated in Section 3.1. Now, to perform the *Galerkin projection* [55, Sect. 2.2.4] onto the POD subspace of dimension $k \leq m$, which is a special case of weighted residual methods, first the function $\boldsymbol{W}$ hast to be substituted by a linear combination of POD modes, i.e.,

$$\boldsymbol{W}(t; \boldsymbol{p}) = \sum_{j=1}^{k} a_j(t; \boldsymbol{p}) \boldsymbol{U}^j, \tag{3.13}$$

and inserted into (2.8) to obtain

$$\sum_{j=1}^{k} \frac{d}{dt} a_j(t;\boldsymbol{p})\boldsymbol{U}^j + \Omega^{-1}\mathbf{res}\Big(\sum_{j=1}^{k} a_j(t;\boldsymbol{p})\boldsymbol{U}^j\Big) = \mathbf{0} \tag{3.14}$$

with unknown functions $a_j(t;\boldsymbol{p})$, $j = 1,\dots,k$, called *POD coefficients*. Finally, the Galerkin projection of (2.8) onto the POD modes is evaluated as

$$\Big\langle \boldsymbol{U}^l, \sum_{j=1}^{k} \frac{d}{dt} a_j(t;\boldsymbol{p})\boldsymbol{U}^j + \Omega^{-1}\mathbf{res}\Big(\sum_{j=1}^{k} a_j(t;\boldsymbol{p})\boldsymbol{U}^j\Big)\Big\rangle = 0, \quad l = 1,\dots,k. \tag{3.15}$$

Due to the orthonormality of the POD modes, a low-dimensional dynamical model of nonlinear ODEs is obtained:

$$\frac{d}{dt} a_j(t;\boldsymbol{p}) + (\boldsymbol{U}^l)^T\Omega^{-1}\mathbf{res}\Big(\sum_{j=1}^{k} a_j(t;\boldsymbol{p})\boldsymbol{U}^j\Big) = 0, \quad l = 1,\dots,k. \tag{3.16}$$

The steady-state solutions of equations (3.16) satisfy the equations

$$(\boldsymbol{U}^l)^T\Omega^{-1}\mathbf{res}\Big(\sum_{j=1}^{k} a_j(\boldsymbol{p})\boldsymbol{U}^j\Big) = 0, \quad l = 1,\dots,k. \tag{3.17}$$

However, the complexity of evaluating the nonlinear term $\mathbf{res}(\cdot)$ remains as costly as before and is a drawback of this method for nonlinear real time applications. Remedy is given by missing point estimation (MPE) [7, 8] or DEIM [25], which is the discrete variant of the empirical interpolation method (EIM) [12]. The key contribution of MPE is to evaluate the nonlinear term over a subset of the spatial domain to speed up the online computations. DEIM by contrast provides a subspace approximation to the nonlinear term by using an additional POD basis. However, the latter is not applicable to steady CFD simulations, because reaching the steady-state means that the residual, i. e. the nonlinear term, vanishes (see equation (2.10)).

3.3. Proper orthogonal decomposition with interpolation

Assuming that the m linearly independent steady-state snapshots $\boldsymbol{W}^1,\dots,\boldsymbol{W}^m$ depend on parameter vectors $\boldsymbol{p}^1,\dots,\boldsymbol{p}^m \in \mathbb{R}^d$, i. e. $\boldsymbol{W}^i = \boldsymbol{W}(\boldsymbol{p}^i)$ for $i = 1,\dots,m$, the POD coupled with an interpolation method can be used to predict a solution $\boldsymbol{W}^*$ at a new location $\boldsymbol{p}^*$ in the parameter space [21]. Here, the explicit projection phase is avoided, making the models independent of the full-order equations, which may thus be constructed given any black-box solver. Moreover, these interpolation-based ROMs are outstandingly cheap to evaluate.

Let the POD modes $\boldsymbol{U}^1, \dots, \boldsymbol{U}^m$ of the snapshot matrix $W = (\boldsymbol{W}^1, \dots, \boldsymbol{W}^m)$ with $\text{rank}(W) = m$ be given by (3.2). By means of (3.3), each snapshot $\boldsymbol{W}^i$ can be expressed in terms of a linear combination of the POD modes with coefficients $a^i_j := \sigma_j V^j_i$:

$$\boldsymbol{W}^i = \sum_{j=1}^{m} a^i_j \boldsymbol{U}^j = (\boldsymbol{U}^1, \dots, \boldsymbol{U}^m) \cdot \boldsymbol{a}^i, \quad \boldsymbol{a}^i = (a^i_1, \dots, a^i_m)^T. \tag{3.18}$$

The procedure of computing approximations $\boldsymbol{W}^*$ at any untried parameter configuration $\boldsymbol{p}^*$ within the (reduced) POD subspace $\text{span}\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^k\}$, $k \leq m$, is as follows [21, 56]:

1. Apply POD to the data matrix W to obtain the corresponding POD modes $\boldsymbol{U}^i$ and POD coefficients a^i_j, whereby $i = 1, \dots, m$, $j = 1, \dots, k$. Due to the relation between the snapshots $\boldsymbol{W}^i$ and their coefficient vectors $\boldsymbol{a}^i = (a^i_1, \dots, a^i_k)^T$, it holds that at the sampled parameter combinations

   $$\boldsymbol{W}(\boldsymbol{p}^i) = \boldsymbol{W}^i, \quad \Rightarrow \quad \boldsymbol{a}^i = \boldsymbol{a}(\boldsymbol{p}^i). \tag{3.19}$$

   Hence, the values of the unknown function $\boldsymbol{a} : \boldsymbol{p} \mapsto \boldsymbol{a}(\boldsymbol{p})$ are known at the sample points $\boldsymbol{p}^i$, $i = 1, \dots, m$. For a fixed POD subspace $\text{span}\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^k\}$, the approximation of a snapshot $\boldsymbol{W}(\boldsymbol{p}^i)$ is given by the orthogonal projection $\bar{U}\bar{U}^T \boldsymbol{W}(\boldsymbol{p}^i)$ onto the POD subspace, where $\bar{U} = (\boldsymbol{U}^1, \dots, \boldsymbol{U}^k)$. If the parametric dependency $\boldsymbol{p} \mapsto \boldsymbol{W}(\boldsymbol{p})$ is smooth, then so is $\boldsymbol{p} \mapsto \bar{U}\bar{U}^T \boldsymbol{W}(\boldsymbol{p})$. Hence, the POD coefficients vector $\boldsymbol{a}(\boldsymbol{p})$ with $a_j(\boldsymbol{p}) = \langle \boldsymbol{W}(\boldsymbol{p}), \boldsymbol{U}^j \rangle$, $j = 1, \dots, k$, depends smoothly on $\boldsymbol{p}$.

2. Thus, interpolation can be used to obtain a new coefficients vector $\boldsymbol{a}^* = \boldsymbol{a}(\boldsymbol{p}^*) \in \mathbb{R}^k$ for any new parameter configuration $\boldsymbol{p}^*$, where the sample data set is now taken to be $\{(\boldsymbol{p}^i, \boldsymbol{a}^i)\}_{i=1}^m$.

3. The prediction of the corresponding solution $\boldsymbol{W}^* = \boldsymbol{W}(\boldsymbol{p}^*) \in \text{span}\{\boldsymbol{U}^1, \dots, \boldsymbol{U}^k\}$, is then given by

   $$\boldsymbol{W}^* = \sum_{j=1}^{k} a^*_j \boldsymbol{U}^j. \tag{3.20}$$

The above method will be here referred to as POD+I.

3.4. Proper orthogonal decomposition with residual optimization

Another method to determine the coefficients of the POD expansion is to solve an optimization problem. Once more consider a set of m linearly independent steady-state snapshots $W = (\boldsymbol{W}^1, \dots, \boldsymbol{W}^m)$ obtained by solving the equations (2.8) along with the corresponding POD modes $\boldsymbol{U}^1, \dots, \boldsymbol{U}^m$. The objective is to choose the coefficients a_j of the (reduced) POD basis expansion

$$\boldsymbol{W}(\boldsymbol{a}) = \sum_{j=1}^{k} a_j \boldsymbol{U}^j, \quad \boldsymbol{a} = (a_1, \dots, a_k), \quad k \le m, \tag{3.21}$$

in such a way that they minimize a given objective function $f : \mathbb{R}^k \to \mathbb{R}$, i. e.

$$\min_{\boldsymbol{a} \in \mathbb{R}^k} f(\boldsymbol{W}(\boldsymbol{a})). \tag{3.22}$$

LeGresley and Alonso [53] proposed a method to calculate approximate flow solutions within the POD subspace by taking the CFD residual into account. Exploiting the cell-wise residual terms $\text{res}_{\cdot,i}(\boldsymbol{W})$ of equations (2.8) or (2.9), they define the dimensionless POD residual as

$$\text{res}_{\cdot,i}^{POD}(\boldsymbol{W}) = 2 + \frac{\text{res}_{\cdot,i}^{+}(\boldsymbol{W})}{\text{res}_{\cdot,i}^{-}(\boldsymbol{W})} + \frac{\text{res}_{\cdot,i}^{-}(\boldsymbol{W})}{\text{res}_{\cdot,i}^{+}(\boldsymbol{W})}, \tag{3.23}$$

where $\text{res}_{\cdot,i}(\boldsymbol{W})$ is splitted into the positive and negative contributions res^+ and res^- in each cell, respectively. Substituting equation (3.21) into (3.23), the objective function f was chosen to be

$$f(\boldsymbol{W}(\boldsymbol{a})) := \sum_{\tilde{n}} (\text{res}_{\tilde{n}}^{POD}(\boldsymbol{W}(\boldsymbol{a})))^2, \tag{3.24}$$

where the summation, $\tilde{n}$, is over a given set of cells in the domain.

Following this approach, Zimmermann et al. [95] determined the coefficients of the reduced order POD solution by minimizing the CFD flux residual $\mathbf{res}(\boldsymbol{W})$ directly:

$$\min_{\boldsymbol{a} \in \mathbb{R}^k} \|\Omega^{-1} \mathbf{res}(\boldsymbol{W}(\boldsymbol{a}))\|_{L_2}^2 = \sum_{j=1}^{n_v} \sum_{i=1}^{n_g} \frac{1}{vol_i} (\text{res}_{j,i}(\boldsymbol{W}(\boldsymbol{a})))^2, \tag{3.25}$$

whereby $f(\boldsymbol{W}(\boldsymbol{a})) := \|\Omega^{-1} \mathbf{res}(\boldsymbol{W}(\boldsymbol{a}))\|_{L_2}^2$ and $\|\boldsymbol{u}\|_{L_2}^2 = (\sqrt{\langle \boldsymbol{u}, \boldsymbol{u} \rangle_{L_2}})^2 = \boldsymbol{u}^T \Omega \boldsymbol{u}$ is the discrete L_2 norm associated with the computational domain of a vector $\boldsymbol{u} \in \mathbb{R}^n$. This ansatz was also modified in [95] to incorporate aerodynamic constraints like, e. g., target lift and target drag.

Since the solutions of equation (3.23) and (3.25) lie inside the (reduced) POD subspace, both cannot be expected to feature a zero residual in general. However, (3.25) provides the best possible approximation to a CFD solution within the POD subspace in a least-squares average sense.

Remark. A starting solution for the optimization problems 3.24 and 3.25 can be obtained by POD+I.

Definition of the residual. The perfomance of the least squares minimization (3.25) strongly depends on how the residual is stated. Employing the *non-descriptor form* of the governing system of ODEs (see eq. (2.8)) or the *descriptor form* $\Omega \frac{d}{dt} \boldsymbol{W} + \mathbf{res}(\boldsymbol{W}) = \mathbf{0}$, the common vectors of the flux residuals for steady CFD in non-descriptor and descriptor form are given by $\Omega^{-1}\mathbf{res}(\boldsymbol{W})$ and $\mathbf{res}(\boldsymbol{W})$, respectively. The former residual scales the residual with the inverse of the cell volumes of the CFD mesh to maximize the accuracy of the computed results in the boundary layer regions where the spatial discretization is the finest. This is preferable, if the problem of interest contains the boundary layer. The latter residual measures the unbalance of the quantities that must be conserved directly and is the standard form for the residual. However, the non-descriptor residual evaluated with respect to the L_2 metric given by equation (3.25) has been advocated in the nonlinear CFD context in [92, 94] and found adventageous in [92] and [5]. In the latter, the non-descriptor residual (3.25) is recoverd as a "hybrid form" with respect to the Euclidean metric (see [93, Appendix A]) and shown to be beneficial for unsteady CFD problems.

CHAPTER 4

Reduced-order modeling via manifold learning

The linear nature of POD makes the method attractive but also is the source of its restriction. To overcome this limitation, a ROM based on ML, or, more generally, DR, is derived in this chapter. One of the most popular ML methods, namely Isomap [85], is employed to extract low-dimensional structures hidden in a given high-dimensional data set $W = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\} \subset \mathbb{R}^n$. Exploiting the theory of Riemannian geometry, Isomap approximates the geodesic distances between each pair of data vectors $\boldsymbol{W}^i$ and $\boldsymbol{W}^j$ and constructs a set of corresponding low-dimensional vectors $Y = \{\boldsymbol{y}^1, \dots, \boldsymbol{y}^m\} \subset \mathbb{R}^d$, $d < n$, whose pairwise Euclidean distances equal the approximated geodesic distances. Based on the set of low-dimensional vectors Y, a mapping from the low-dimensional space to the high-dimensional space is derived. Coupled with an interpolation model formulated between the parameter space $\mathcal{P}$ and the low-dimensional space spanned by the set Y, a ROM is obtained which is capable of predicting full-order solutions at untried parameter combinations $\boldsymbol{p} \in \mathcal{P}$. Furthermore, another back-mapping from the low-dimensional space to the high-dimensional space based on residual optimization is derived. Its objective is to obtain a CFD-enhanced prediction by minimizing the discretized flux residual of the interpolated solution.

The chapter is organized as follows: first, an introduction to manifolds and Riemannian geomerty is outlined in Section 4.1. In Section 4.2, the definition of ML along with a brief overview of various methods of this class is given. Afterwards, the ML method Isomap is introduced in Section 4.3 followed by the derivation of the general back-mapping in Section 4.4, which is exploited to deduce two kinds of ROMs in Sections 4.5 and 4.6. Section 4.7 finally gives an idea of additional possible applications of Isomap and Isomap+I.

4.1. Manifolds and geodesic distances

Manifolds are an abstraction of the idea of smooth surfaces in Euclidean space and, in the simplest term, they are spaces which consist of smooth patches, that look locally like a Euclidean space. Consider the sphere depicted in Figure 4.1. The extrinsic dimension is three as each point on the sphere may be addressed by its (x, y, z)-coordinates, but the sphere can be parameterized by only two variables. Furthermore, a small neighborhood of a point on the sphere looks locally like a two-dimensional Euclidean space. This intuition motivates the formal definition of manifolds. But before the definition is given, prior knowledge of topology is required.

4.1 Definition. A *topological space* $\mathcal{M}$ is a set $\mathcal{M}$ together with a collection of subsets, called *open sets*, such that the empty set $\emptyset$, the whole set $\mathcal{M}$ and the union of any collection of open sets is itself an open set. The latter must also hold for the intersection of any finite collection of open sets.

4.2 Definition. A function $f : \mathcal{M} \to \mathcal{N}$ between two topological spaces $\mathcal{M}$ and $\mathcal{N}$ is called *continuous* if for every open subset $N \subset \mathcal{N}$ the preimage $f^{-1}(N) = \{m \in \mathcal{M} : f(m) \in N\}$ is an open subset of $\mathcal{M}$.

4.3 Definition. Let $\mathcal{M}$ be a topological space, $U \subset \mathcal{M}$ and $\Omega \subset \mathbb{R}^d$ open subsets of $\mathcal{M}$ and $\mathbb{R}^d$, respectively. A *homeomorphism* $h : U \to \Omega$ is a continuous, bijective mapping between U and Ω, whose inverse also is continuous.

4.4 Definition. Let $\mathcal{M}$ be a topological space. $\mathcal{M}$ is a *Hausdorff space*, if for all $x, y \in \mathcal{M}$, $x \neq y$, there exist open sets $U, V \subset \mathcal{M}$ with $x \in U$ and $y \in V$, so that $U \cap V = \varnothing$.

4.5 Definition. A $\mathcal{C}^q$*-atlas*, $q \in \mathbb{N} \cup \{\infty\}$, on a Hausdorff space $\mathcal{M}$ is given by an open covering U_i, $i \in \mathcal{I}$, of $\mathcal{M}$ and a family of homeomorphism $h_i : U_i \to \Omega_i$, where Ω_i are open subsets of $\mathbb{R}^d$, such that for any i and j in $\mathcal{I}$, the homeomorphism

$$h_j \circ h_i^{-1} : h_i(U_i \cap U_j) \to h_j(U_i \cap U_j)$$

is in fact a $\mathcal{C}^q$*-diffeomorphism*, i. e. a homeomorphism whose inverse is also $\mathcal{C}^q$-differentiable. The $(U_i, h_i)_{i \in \mathcal{I}}$ are called *charts* for $\mathcal{M}$.

With the additional structure given by an atlas, the definition of a differentiable manifold is as follows:

4.6 Definition. A connected Hausdorff topological space $\mathcal{M}$ is called a $\mathcal{C}^q$ *differentiable manifold* of dimension d, if there exists a $\mathcal{C}^q$-atlas on $\mathcal{M}$.

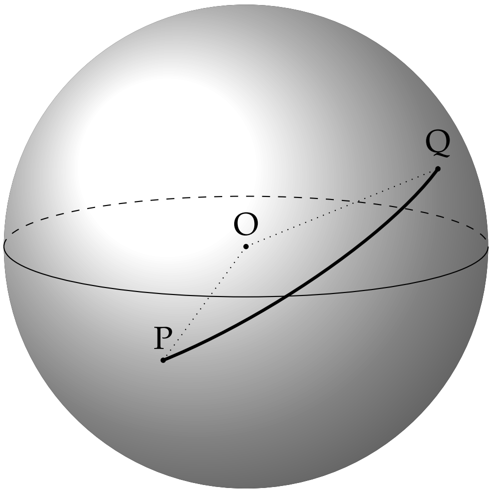


Figure 4.1. A sphere as an example of a two-dimensional submanifold. The geodesic is illustrated by the solid line between the points P and Q.

"Charts allow to generalize to manifolds the classical notions of differential calculus, such as vector fields and differential forms"[38, Chapter 1], by carrying the differentiable structure from the open subsets of $\mathbb{R}^d$ onto the corresponding neighborhoods in $\mathcal{M}$.

The simplest example of a (linear) manifold is the $\mathbb{R}^d$ itself. Further (nonlinear) examples are smooth curves such as circles or parabolas or smooth surfaces such as spheres (shown in Fig. 4.1), paraboloids or tori.

4.7 Example. The unit sphere $S^2 := \{(x,y,z) \in \mathbb{R}^3 \mid \|(x,y,z)\|_2 = 1\}$ in Fig. 4.1 is a two-dimensional differential manifold of class $\mathcal{C}^\infty$. For example, a common choice for the atlas is a covering by the sets

$$\begin{aligned} U_1 &:= S^2 \setminus \{(0,0,1)\}, \\ U_2 &:= S^2 \setminus \{(0,0,-1)\} \end{aligned}$$

along with the charts

$$\begin{aligned} &h_1 : U_1 \to \mathbb{R}^2, \\ &(x,y,z) \mapsto \left(\tfrac{x}{1-z}, \tfrac{y}{1-z}\right) \end{aligned}$$

and

$$\begin{aligned}h_2 : U_2 &\to \mathbb{R}^2,\\ (x,y,z) &\mapsto (\tfrac{x}{1+z}, \tfrac{y}{1+z}).\end{aligned}$$

These charts are called the stereographic projections. Since $h_1(U_1 \cap U_2) = h_2(U_1 \cap U_2) = \mathbb{R}^2 \setminus \{(0,0)\}$ and $h_1 \circ h_2^{-1}(y) = \frac{y}{\|y\|^2} = h_2 \circ h_1^{-1}(y)$ for all $y \in \mathbb{R}^2 \setminus \{(0,0)\}$, we have a differentiable atlas for S^2.

Considering ML, which deals with d-dimensional manifolds lying within a Euclidean space of dimension $d+n$, it is sufficient to restrict the considerations to *submanifolds* $\mathcal{M} \subset \mathbb{R}^{d+n}$ [48]:

4.8 Definition. Let n, $d \in \mathbb{N}$, and $q \in \mathbb{N} \cup \{\infty\}$ be given. A subset $\mathcal{M} \subset \mathbb{R}^{d+n}$ is a d-dimensional *$\mathcal{C}^q$-differentiable submanifold* of $\mathbb{R}^{d+n}$, if for all $x \in \mathcal{M}$, there exist open neighborhoods $U,V \subset \mathbb{R}^{d+n}$, $x \in U$, and a $\mathcal{C}^q$-diffeomorphism $h : U \to V$ such that $h(U \cap \mathcal{M}) = V \cap (\mathbb{R}^d \times \{0\})$. The *charts* are given by $h|_{U\cap\mathcal{M}}$ and the *atlas* is the set of all charts as defined before.

More intuitively spoken, a d-dimensional differentiable submanifold of $\mathbb{R}^{d+n}$ is a subset of $\mathbb{R}^{d+n}$, which in essence locally looks like a $\mathbb{R}^d$ embedded in a $\mathbb{R}^{d+n}$.

4.9 Example. The unit sphere S^2 is a two-dimensional submanifold of $\mathbb{R}^3$. To proof this, it has to be shown that for all $\hat{p} \in S^2$, there exists a diffeomorphism $h : U \to V$ such that $h(U \cap S^2) = V \cap (\mathbb{R}^2 \times \{0\})$. Let $p = (0,0,1) \in S^2$ be the north pole of the sphere and

$$\begin{aligned}U &:= \{(x,y,z) \in \mathbb{R}^3 : z > 0,\ \|(x,y)\|_2 < 1\}\\ V &:= \{(x,y,z) \in \mathbb{R}^3 : z > -\sqrt{1-x^2-y^2},\ \|(x,y)\|_2 < 1\}\end{aligned}$$

Obviously it holds that $p \in U$. Furthermore, the map

$$h : U \to V, \quad (x,y,z) \mapsto (x,y,z-\sqrt{1-x^2-y^2}),$$

is a $\mathcal{C}^\infty$-diffeomorphism, where its inverse is given by

$$h((x,y,z))^{-1} = (x,y,z+\sqrt{1-x^2-y^2})$$

for $(x,y,z) \in V$. Since

$$U \cap S^2 = \{(x,y,\sqrt{1-x^2-y^2}) : x,y \in \mathbb{R},\ \|(x,y)\| < 1\}$$

it holds that $h(U \cap S^2) = V \cap (\mathbb{R}^2 \times \{0\})$. For an arbitrary point $\hat{p} \in S^2$, there exists a transformation $T \in O_3 = \{A \in \mathbb{R}^{3\times 3} : A^T A = I\}$ such that $T\hat{p} = p$. Hence $U' := T^{-1}(U) \subset \mathbb{R}^3$ is an open subset, $\hat{p} \in U'$ and $h \circ T : U' \to V$ is a $\mathcal{C}^\infty$-diffeomorphism. Since $T(S^2) = S^2$ it holds that $(h \circ T)(U' \cap S^2) = h(U \cap S^2) = V \cap (\mathbb{R}^2 \times \{0\})$.

A local "linear" approximation to a differentiable submanifold $\mathcal{M} \subset \mathbb{R}^{d+n}$ at $p \in \mathcal{M}$ is given by the following definition that is related to the local linear approximation to a function by its first-order derivative:

4.10 Definition. Let $\mathcal{M}$ be a d-dimensional differentiable submanifold of $\mathbb{R}^{d+n}$. The *tangent space* $T_p\mathcal{M}$ to $\mathcal{M}$ at $p \in \mathcal{M}$ is defined as the set of all velocity vectors $v = \frac{d\gamma}{dt}(0)$ of differentiable curves $\gamma : (-\varepsilon, \varepsilon) \to \mathcal{M}$ with $\gamma(0) = p$.

The vector v is well-defined, because the differentiation of $\gamma(t)$ occurs in the Euclidean space [38, 1.21]. Note that, if $\mathcal{M} \subset \mathbb{R}^{d+n}$, then $T_p\mathcal{M}$ is a d-dimensional linear subspace of $\mathbb{R}^{d+n}$.

To measure angles and lengths of curves on a differentiable manifold, it has to be endowed with a smooth inner product g_p on each tangent space $T_p\mathcal{M}$, $p \in \mathcal{M}$.

4.11 Definition. A differentiable manifold $\mathcal{M}$ endowed with a *Riemannian metric* $g := \{g_p\}_{p\in\mathcal{M}}$ of inner products g_p, is called a *Riemannian manifold* $(\mathcal{M}, g)$.

There exists at least one Riemannian metric on any differentiable manifold $\mathcal{M}$, which is connected and countable at infinity [38, 2.2]. Furthermore, each submanifold $\mathcal{M}$ of a Euclidean space $\mathbb{R}^{d+n}$ is canonically equipped with the Riemannian metric g defined by restricting the ambient scalar product of $\mathbb{R}^{d+n}$ to each tangent space $T_p\mathcal{M}$ [38, 2.8]. A distance on a Riemannian manifold $(\mathcal{M}, g)$, which induces the topology of $\mathcal{M}$ and turns the manifold into a metric space, is defined by the infimum of the lengths of all piecewise $\mathcal{C}^1$-curves between two points [38, 2.91].

4.12 Definition. Piecewise $\mathcal{C}^1$-curves γ on $\mathcal{M}$, which join two points along the (locally) shortest path and which are parameterized propotionally to arc length, i. e. $\|\frac{d\gamma}{dt}(t)\| \equiv$ const., are called *geodesics* [38, 2.97].

Geodesics are a generalization of the notion of a "straight line" in the Euclidean space to curved spaces. In the case of the sphere depicted in Fig. 4.1, where the geometry is well-known, the geodesic joining P and Q is the circular arc between P and Q and implicitly given by the radius of the sphere and the angle between the two line segments $\overline{OP}$ and $\overline{OQ}$. For more complex or even unknown geometries, determining the geodesic between two points, or only the length of it, is a highly challenging task.

Manifolds and Riemannian geometry are well-studied in the field of mathematics, however some of the above definitions are formulated more generally. For a more detailed introduction to manifolds and Riemannian geometry see [38] or [82].

4.2. Manifold learning / dimensionality reduction

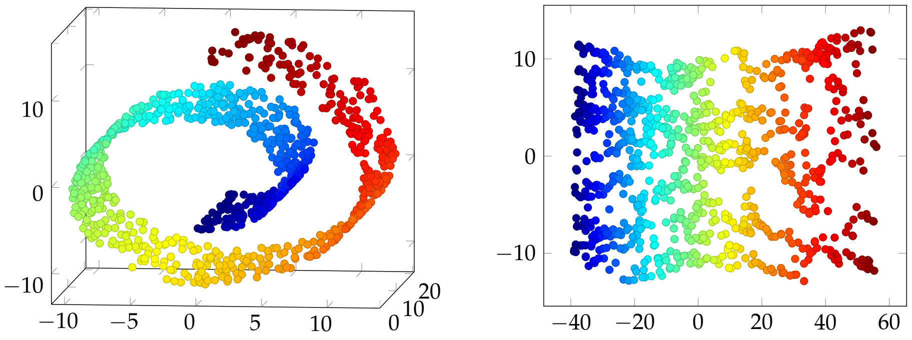
High dimensional real world data tends to concentrate in the spatial proximity of nonlinear low-dimensional submanifolds of the data space. This hypothesis, called the *manifold hypothesis*, is the basis of the data driven methods termed manifold learning [61]. "The basic objective of most nonlinear ML techniques, or, more generally, DR techniques, is to solve the so-called embedding problem [88, 23, 17], which can be stated as follows:

> For a given data set $W = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\} \subset \mathcal{W} \subset \mathbb{R}^n$ sampled from an unknown submanifold $\mathcal{W}$ of the Euclidean space $\mathbb{R}^n$ with intrinsic dimensionality $\dim \mathcal{W} = d < n$ (usually, $d \ll n$ is to be expected), find an embedding mapping
>
> $$\boldsymbol{h} : \mathcal{W} \to \mathbb{R}^d, \quad \boldsymbol{h}(\boldsymbol{W}^i) = \boldsymbol{y}^i \in \mathbb{R}^d, \tag{4.1}$$
>
> which preserves the geometry of the data set W as much as possible, so that the low-dimensional data set $Y = \{\boldsymbol{y}^1, \dots, \boldsymbol{y}^m\}$ is a good representation of the high-dimensional data set W."(Franz et al. [36])

Thus, ML methods try to recover in a meaningful way low-dimensional structures hidden in high-dimensional data, which is assumed to lie on or close to a *smooth embedded submanifold* $\mathcal{W} \subset \mathbb{R}^n$ covered by a *single* coordinate chart $\boldsymbol{h}$. "In general, ML/DR is an ill-posed problem, because neither the geometry of the data nor the *intrinsic dimensionality* [, which is the minimum number of parameters needed to account for the observed properties of the data [37],] is known."(Franz et al. [36])

A simple example often encountered in the literature of manifold learning is the embedding of the swiss roll, a two-dimensional manifold embedded in $\mathbb{R}^3$. Figure 4.2 shows the swiss roll and its two-dimensional embedding "learned" by the method Isomap, which is a specific ML algorithm to be discussed later in Section 4.3. Notice that nearby points as measured on the embedded submanifold (see Figure 4.2(a)) are also close-by in the two-dimensional embedding space (see Figure 4.2(b)).

Classical techniques like PCA [80] or multidimensional scaling (MDS) [58] work for almost linear submanifolds $\mathcal{W} \subset \mathbb{R}^n$, but fail as soon as the submanifold becomes nonlinear [85]. Therefore, several new nonlinear approaches for ML/DR have been



(a) Swiss roll data set consisting of 1000 data points.

(b) Two-dimensional embedding of the swiss role data set.

Figure 4.2. Example of a two-dimensional embedding of a manifold with intrinsic dimension two, which lies in the Euclidean space $\mathbb{R}^3$. The embedding is calculated by Isomap (see Section 4.3).

proposed in the last decade, which preserve different geometrical features of a manifold. They can be subdivided into three parts: local techniques, global techniques and techniques that globally align linear models.

Local approaches like LLE [70] or Laplacian Eigenmaps [14] attempt to preserve the local geometry of the data by preserving properties of small neighborhoods around the datapoints. In fact, they aspire to map nearby points on the manifold to nearby points in the low-dimensional representation. By preserving local properties, these methods assert to retain the global structure of the data as well.

In contrast, global techniques like Isomap [85] or multilayer autoencoders [44] attempt to preserve global properties of the data. For example, Isomap tries to map nearby points on the manifold to nearby points in the low-dimensional space, and faraway points to faraway points, which tends to give a more faithful representation of the data's global structure than when retaining only local properties [27]. Training a multilayer autoencoder network, however, leads to a network in which the middle hidden layer gives a low-dimensional representation of the data which retains as much information as possible.

Combinations of the previous types are given by, e. g., locally linear coordination (LLC) [84] and manifold charting [20]. Both techniques first compute a number of locally linear models and then perform a global alignment of these models.

A comprehensive overview and comparative review of various approaches to ML are given by Van der Maaten, Postma, and Van den Herik [88].

4.3. Manifold learning by Isomap

Many well-known curves and surfaces, e.g. all spheres $S^{n-1} \subset \mathbb{R}^n$, are specified as submanifolds of a Euclidean space and inherit a Riemannian metric from their ambient space. Assuming that the manifold hypothesis is true, the unknown submanifold $\mathcal{W}$ can also be regarded as a Riemannian manifold equipped with the Riemannian metric inherited by the ambient space (see Section 4.1). Hence, the notion of the distance between two points $\boldsymbol{W}^i, \boldsymbol{W}^j \in \mathbb{R}^n$ might be given by the infimum of the length of all piecewise $\mathcal{C}^1$ curves between two points $\boldsymbol{W}^i, \boldsymbol{W}^j \in \mathcal{W}$ on the manifold, i. e. by the length of the corresponding *geodesic*. A nonlinear ML method that exploits the aforementioned concept and approximately solves the embedding problem is Isomap [85]. It is based on metric MDS [58] and perhaps the best known and most applied method of all available ML methods [23]. Furthermore, it is intuitive, well-understood and supported by theoretical results [18]. "The metric MDS method preserves the inter-point distances of the input sample data points $W = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\}$ considered as vectors in $\mathbb{R}^n$, which may be far off the geodesic distances of the data points considered as points in $\mathcal{W}$, see Figure 4.3 for a schematic illustration using *Euclidean* distances."(Franz et al. [36]) In contrast, Isomap attempts to preserve the geodesic distances (see Section 4.1) between all pairs of points as a feature of the underlying geometry by providing an (almost) isometric mapping:

4.13 Definition. Let (X, d_x) and (Y, d_y) be metric spaces with metrics d_x and d_y. A map $\boldsymbol{f} : X \to Y$ is called an *isometry*, if for any $\boldsymbol{a}, \boldsymbol{b} \in X$ it holds that $d_y(\boldsymbol{f}(\boldsymbol{a}), \boldsymbol{f}(\boldsymbol{b})) = d_x(\boldsymbol{a}, \boldsymbol{b})$.

In the case of Isomap, the metric spaces are given by $(\mathcal{W}, \|\cdot\|_{geo})$ and $(\mathbb{R}^d, \|\cdot\|_2)$, and an isometric "chart" $\boldsymbol{h} : \mathcal{W} \to \mathbb{R}^d$ for a discrete set of points $W \subset \mathcal{W}$ is sought-after. Here, $\|\cdot\|_{geo}$ denotes the infimum of the length of the all piecewise $\mathcal{C}^1$ curves between two points $a, b \in \mathcal{W}$ on the manifold as introduced in Section 4.1. An example of an embedding calculated by Isomap is shown in Figure 4.2.

"Since the geometry of the data manifold is unknown a priori, an approximation of the geodesic distances is required before MDS can be applied to the data set in a geometrically meaningful way."(Franz et al. [36])

Remark. Due to the intuitive approach of Isomap, extensions like, among others, Isomap for conformal embeddings (C-Isomap) [27] and Isomap with landmark points (L-Isomap) [28] have been published. The former method preserves angles but not lengths, whereas the latter exploits only a subset of the available data to compute the embedding to reduce the costs of MDS. Since the accurate lengths, or, more precisely, the accurate distances of nearby points are crucial for the back-mapping (see

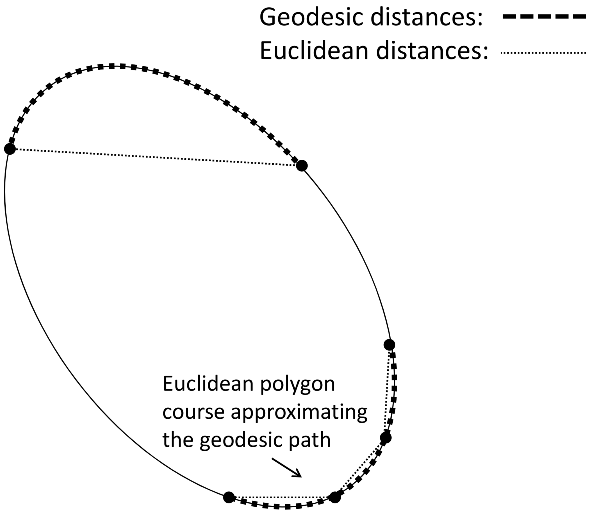


Figure 4.3. "Schematic representation of geodesic distances (shortest paths on the manifold) vs. Euclidean distances measured in the ambient space, ignoring the manifold structure."(Franz et al. [36])

Section 4.4) and the number of available data points in aerodynamic applications is small compared to common applications of Isomap, both extensions are not of interest for our purposes.

4.3.1. Approximating geodesic distances

Let $W = \{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^m\}$ be a set of full-order snapshots sampled from an unknown submanifold $\mathcal{W} \subset \mathbb{R}^n$. Furthermore, it is assumed that $\mathcal{W}$ is sufficiently smooth so that the geodesics between close-by snapshots are approximately linear. Thus, "[f]or full-order data vectors lying close to each other, it is assumed that the Euclidean distance as measured in the ambient space $\mathbb{R}^n$ provides a good approximation to the intrinsic geodesic distance. Based on this assumption, the idea of Isomap is to estimate the geodesic distance by the length of a Euclidean polygonal curve, where the vertices are chosen as close-by data points, see again Fig. 4.3. Thus, the problem of computing geodesic distances on the manifold is replaced by a graph-theoretical shortest paths problem.

The first step in Isomap is to construct an undirected weighted neighborhood graph $G = (V, E, c)$, whose nodes V are given by the input data snapshots $\{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^m\}$. The edge set E consists of the edges between nearby points selected by a criterion to be specified by the user and the edge weight function c is the Euclidean distance $c(\boldsymbol{W}^i, \boldsymbol{W}^j) := \|\boldsymbol{W}^i - \boldsymbol{W}^j\|_2$. The edge set can be determined either by connecting each point $\boldsymbol{W}^i$ with all points lying within a ball of radius ε [given by the set

$\{\boldsymbol{W} \in W \setminus \{\boldsymbol{W^i}\} \mid \|\boldsymbol{W} - \boldsymbol{W^i}\|_2 < \varepsilon\}$] or by connecting each point with its [Euclidean] k nearest neighbors [...]. Then, the geodesic distance between any pair of snapshots is approximated as the length of the corresponding shortest path within the neighborhood graph G[, e. g. calculated via the Floyd-Warshall algorithm [32]]. Algorithm 4.1 details this approximation.

For data manifolds, which are isometric to a convex domain of the Euclidean space and fulfill further assumptions on the sampling and the neighborhood graph, it can be shown that this approximation asymptotically converges to the geodesic distances when the number of data sample points tends to infinity and therefore that Isomap asymptotically recovers the true geometric structure of the manifold [18]"(Franz et al. [36]) This result holds for both types of nearest neighbor queries.

Remark. In an aerodynamic framework, the data vectors $\boldsymbol{W}^i$ may consist of flow quantities like density, velocity and pressure, which widely differ in their magnitude. Since Isomap is based on nearest neighbor queries, each quantity should be scaled to the unit cube $[0,1]^{n_g}$ separately. More precisely, for each variable v, find the minimum and maximum value $\min_v$ and $\max_v$ of all data vectors $\boldsymbol{W}^i$, $i = 1, \ldots, m$, respectively, and set $W^i_j := \frac{W^i_j - \min_v}{\max_v - \min_v}$ for all j belonging to variable v. Otherwise, the algorithm may perform poorly.

4.3.2. Multidimensional scaling

> "Multidimensional Scaling (MDS) is concerned with the problem of constructing a configuration of n points in Euclidean space using information about the distance between the n objects. [...] In general the purpose of MDS is to provide a "picture" which can be used to give a meaningful interpretation of the data. Hopefully, the picture will convey useful information about the relationship between the objects."
>
> *by Mardia, Kent, and Bibby [58, Section 14.1]*

"Using Multidimensional Scaling (MDS), high-dimensional data is mapped to a low-dimensional representation while preserving the pairwise distances between all points of the data set as much as possible. More precisely, given a high-dimensional data set $\boldsymbol{W}^1, \ldots, \boldsymbol{W}^m \in \mathbb{R}^n$ with inter-point distances $d_{ij} = \operatorname{dist}(\boldsymbol{W}^i, \boldsymbol{W}^j)$ collected in a quadratic (possibly non-Euclidean) distance matrix $D = (d_{ij})_{i,j=1,\ldots,m}$, MDS is concerned with determining a *fitted embedding* $\hat{X} \in \mathbb{R}^{d \times m}$, such that the columns $\hat{\boldsymbol{x}}^1, \ldots, \hat{\boldsymbol{x}}^m \in \mathbb{R}^d$ of $\hat{X}$ constitute a data set in $\mathbb{R}^d$ featuring *Euclidean* inter-point distances $\hat{d}_{ij} = \|\hat{\boldsymbol{x}}^i - \hat{\boldsymbol{x}}^j\|_2$ that equal the inter-point distances of the original data,

Algorithm 4.1 "Approximating geodesic distances."(Franz et al. [36])

```
Require: Data set $W = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^m\}$, $k \in \mathbb{N}$
Construct neighborhood graph $G$:
$G \leftarrow (W, E = \emptyset, c)$                                   ▷ undirected weighted graph
for $i = 1$ to $m$ do
    if $k$-nearest neighbors then
        determine Euclidean $k$ nearest neighbors $\boldsymbol{W}^l \in W \setminus \{\boldsymbol{W}^i\}$ of $\boldsymbol{W}^i$
        $N \leftarrow \{\boldsymbol{W}^{l_1}, \dots, \boldsymbol{W}^{l_k}\} \subset W \setminus \{\boldsymbol{W}^i\}$
    if $\varepsilon$-neighborhood then
        determine all neighbors $\boldsymbol{W}^l$ within an $\varepsilon$-ball around $\boldsymbol{W}^i$
        $N \leftarrow \{\boldsymbol{W}^{l_1}, \dots, \boldsymbol{W}^{l_{|NN|}}\} \subset W \setminus \{\boldsymbol{W}^i\}$
    for all $\boldsymbol{W} \in N$ do
        $E \leftarrow E \cup (\boldsymbol{W}^i, \boldsymbol{W})$
return Neighborhood graph $G$

Compute shortest paths in neighborhood graph $G$:
Initialize $m \times m$ distance matrix $D = (d_{ij})_{i,j=1,\dots,m}$ with $d_{ij} := 0$ if $i = j$, $d_{ij} := \infty$
otherwise
for all $e = (\boldsymbol{W}^i, \boldsymbol{W}^j) \in E$ do
    $d_{ij} = d_{ji} \leftarrow c(e)$
for $i, j, k = 1$ to $m$ do
    $d_{ij} \leftarrow \min\{d_{ij}, d_{ik} + d_{kj}\}$                  ▷ Floyd-Warshall
return Approximated geodesic distance matrix $D$
```

i.e., $\hat{d}_{ij} = d_{ij}$[, and is unique up to translation and rotation]. The classical solution to the MDS problem is given in the following theorem:

4.14 Theorem ([58], Thm. 14.2.1). *Let $D = (d_{ij})_{i,j=1,\dots,m} \in \mathbb{R}^{m \times m}$ be a distance matrix and let $\mathbb{1} = (1, \dots, 1)^T \in \mathbb{R}^m$ be the vector with all entries equal to 1. Define the* double centered matrix $B = -\frac{1}{2} H \tilde{D} H$, *where $\tilde{D} = (\tilde{d}_{ij})_{i,j=1,\dots,m} = (d_{ij}^2)_{i,j=1,\dots,m}$ and $H = I_{m \times m} - m^{-1} \mathbb{1}\mathbb{1}^T$ denotes the orthogonal projection onto $\operatorname{span}\{\mathbb{1}\}^\perp$. If B is positive semi-definite and $\operatorname{rank}(B) = d$, then an MDS embedding in the d-dimensional space $\mathbb{R}^d$ is constructed as follows.*

Let $\lambda_1 > \cdots > \lambda_d$ denote the positive eigenvalues of B with corresponding eigenvector matrix $X = (\boldsymbol{x}^1, \dots, \boldsymbol{x}^d) \in \mathbb{R}^{m \times d}$ normalized by $(\boldsymbol{x}^i)^T \boldsymbol{x}^i = \lambda_i$, $i = 1, \dots, d$. Then $\hat{X} := X^T \in \mathbb{R}^{d \times m}$ is a fitted embedding, i.e., the row vectors $\boldsymbol{y}^k := (x_k^1, \dots, x_k^d) \in \mathbb{R}^d$, $k = 1, \dots, m$ of X feature the same inter-point distances as listed in D.

Remark. Theorem 4.14 is valid for all Euclidean distance matrices D, because the corresponding double-centered matrix B is positive semi-definite if and only if D is a Euclidean distance matrix, see [40, eq. (1)]. [Moreover, if D is a Euclidean distance matrix, then there is a close connection between MDS and PCA [58, Section 14.3].]

The above theorem leads to a practical algorithm to find an MDS embedding in a Euclidean space of low dimension $d < n$:

1. Construct the distance matrix D and calculate the double centered matrix B.
2. Solve the eigenvalue problem $B\boldsymbol{v} = \lambda \boldsymbol{v}$ to obtain the d largest eigenvalues $\lambda_1 > \cdots > \lambda_d$ with corresponding eigenvectors $X = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^d)$, which are normalized by $(\boldsymbol{x}^i)^T \boldsymbol{x}^i = \lambda_i$, $i = 1, \ldots, d$ (assuming that the first d eigenvalues are positive).
3. The embedding is given by the vectors $\boldsymbol{y}^k := (x_k^1, \ldots, x_k^d)$, $k = 1, \ldots, m$.

For arbitrary distance matrices the following optimality property is fulfilled:

4.15 Theorem ([58], Thm. 14.4.2)**.** *Let D be an arbitrary distance matrix with corresponding double-centered matrix $B = (b_{ij})_{i,j=1,\ldots,m}$ and let $\hat{B} = (\hat{b}_{ij})_{i,j=1,\ldots,m}$ be the centered inner product matrix for a fitted embedding $\hat{X}$. Then, for fixed d, the discrepancy*

$$\sum_{i,j=1}^{m} (\hat{b}_{ij} - b_{ij})^2 \tag{4.2}$$

between B and $\hat{B}$ is minimized over all embeddings $\hat{X}$ in d dimensions, when $\hat{X}$ is the classical solution to the MDS problem.

As a consequence, MDS yields the best d-dimensional Euclidean embedding of the given data manifold relative to the discrepancy (4.2), even if the the double-centered matrix B corresponding to the approximated geodesic distance matrix D is not positive definite [,i. e., non-Euclidean]."(Franz et al. [36])

The classical solution of MDS as introduced above favors the accurate fit of far away points when the discrepancy of inter-point distances is large (see discrepancy (4.2)). Especially when dealing with large parameter spaces, the maximum discrepancy of inter-point distances between solutions increases and hence the distances between close-by points become inaccurate. Nevertheless, the distances between nearby points are an essential part of the mapping from the reduced-order representation Y to full-dimensional snapshot approximations (see Section 4.4) and therefore accurate distances between close-by points are indispensable. Hence, MDS should yield an embedding $Y = \{\boldsymbol{y}^1, \ldots, \boldsymbol{y}^m\}$ which penalizes larger distances more than smaller distances by imposing additional weights $w_{ij} \in \mathbb{R}$:

$$Y = \underset{\hat{Y} \in \mathbb{R}^{d \times m}}{\arg\min} \sum_{\substack{i,j=1 \\ i \neq j}}^{m} \frac{(\hat{d}_{ij} - d_{ij})^2}{w_{ij}}, \tag{4.3}$$

where $d_{ij} = \operatorname{dist}(\boldsymbol{W}^i, \boldsymbol{W}^j)$ and $\hat{d}_{ij} = \|\hat{\boldsymbol{y}}^i - \hat{\boldsymbol{y}}^j\|_2$. Reasonable choices for the weights are $w_{ij} := d_{ij}$, $w_{ij} := d_{ij}^2$, or, w_{ij} could be set to the number of edges of the shortest path connecting W^i and W^j. The former choice is similar to Sammon's stress [72] and the latter is called edge number-based Isomap (EN-Isomap) in the literature [76]. The corresponding *loss function*

$$f(\hat{Y}) = \sum_{\substack{i,j=1 \\ i \neq j}}^{m} \frac{(\hat{d}_{ij} - d_{ij})^2}{w_{ij}}, \quad \hat{d}_{ij} = \hat{d}_{ij}(\hat{\boldsymbol{y}}^i, \hat{\boldsymbol{y}}^j), \tag{4.4}$$

is minimized by variable alternation [87] similar to as utilized for EN-Isomap in [76]. Fixing all but one variables and optimizing the remaining one results in a subproblem, which is much easier to solve than the problem in its entirety. The variable, which is selected to be fixed, varies during the iterative process.

The stationary points of f are characterized by $\frac{\partial f}{\partial \hat{\boldsymbol{y}}^i} = 0$ for $i = 1, \ldots, m$. It holds that

$$\begin{aligned} 0 &= \frac{\partial f}{\partial \hat{\boldsymbol{y}}^i} && (4.5)\\ &= 2 \sum_{j \neq i} \frac{\partial f}{\partial \hat{d}_{ij}} \cdot \frac{\partial \hat{d}_{ij}}{\partial \hat{\boldsymbol{y}}^i} && (4.6)\\ &= 4 \sum_{j \neq i} \frac{\hat{d}_{ij} - d_{ij}}{w_{ij}} \cdot \frac{\hat{\boldsymbol{y}}^i - \hat{\boldsymbol{y}}^j}{\hat{d}_{ij}} && (4.7)\\ &= 4 \Big(\hat{\boldsymbol{y}}^i \sum_{j \neq i} \frac{1}{w_{ij}} - \sum_{j \neq i} \big(\frac{1}{w_{ij}} \hat{\boldsymbol{y}}^j + \frac{d_{ij}}{w_{ij} \hat{d}_{ij}} (\hat{\boldsymbol{y}}^i - \hat{\boldsymbol{y}}^j) \big) \Big) && (4.8)\\ \Leftrightarrow \hat{\boldsymbol{y}}^i &= \frac{\sum_{j \neq i} \big(\frac{1}{w_{ij}} \hat{\boldsymbol{y}}^j + \frac{d_{ij}}{w_{ij} \hat{d}_{ij}} (\hat{\boldsymbol{y}}^i - \hat{\boldsymbol{y}}^j) \big)}{\sum_{j \neq i} \frac{1}{w_{ij}}}. && (4.9) \end{aligned}$$

Employing the adjustment rule (4.9), variable alternation is performed as stated in Algorithm 4.2. An initial embedding Y for the iterative process is given by the classical MDS solution as stated in Theorem 4.14.

Remark. A more common way to solve the optimization problem (4.3) is using majorization [26]. However, in our test cases the variable alternation strategy outperformed the SMACOF (Scaling by MAjorizing a COmplicated Function) strategy.

Algorithm 4.2 Variable Alternation.

```
Require: Initial embedding $Y = \{\boldsymbol{y}^1, \dots, \boldsymbol{y}^m\}$, tolerance $\delta$, maximum iterations $l$
$\hat{l} \leftarrow 0$
repeat
    $\hat{l} \leftarrow \hat{l} + 1$
    $Y_{old} \leftarrow Y$
    for $i = 1, \dots, m$ do                                   ▷ alternate $Y$
        $\boldsymbol{y}^i \leftarrow \big(\sum_{j \neq i} \big(\frac{1}{d_{ij}^2} \boldsymbol{y}^j + \frac{d_{ij}}{d_{ij}^2 \hat{d}_{ij}} (\boldsymbol{y}^i - \boldsymbol{y}^j)\big)\big) / \sum_{j \neq i} \frac{1}{d_{ij}^2}$
until $\max_{i,j}(|(Y - Y_{old})_{ij}|) < \delta$ or $\hat{l} \geq l$
return $Y$
```

4.3.3. Isomap and the choice of its parameters

"After the preparatory work in the previous sections, Isomap can be reduced to the following two steps:

1. Approximate the geodesic distances between all pairs of full-order data points to build the corresponding distance matrix D.

2. Apply possibly weighted MDS to D to obtain a low dimensional embedding, which preserves the geometry of the underlying high dimensional data set.

The number of nearest neighbors k or the radius of the ε-ball to construct a neighborhood graph and the dimension d of the low-dimensional space have to be specified to construct an isometric embedding. The optimal choice of these parameters is problem dependent and the success of Isomap depends largely on the selected values [9]."(Franz et al. [36]) Assuming that the flow solutions to the Navier-Stokes equations exist and depend smoothly on a parameter configuration $\boldsymbol{p} \in \mathbb{R}^{\tilde{d}}$, the intrinsic dimensionality of the manifold is know a priori, namely $\tilde{d}$, and thus the dimension d of the embedding space is set to $\tilde{d}$. A suitable k- or ε-neighborhood for Isomap can be determined by minimizing, e. g., the *Kruskal stress* [78], which is given by

$$s(\tilde{k}) = \sqrt{\frac{\sum_{i,j=1}^{m} (d_{ij}(\tilde{k}) - \|\boldsymbol{y}^i(\tilde{k}) - \boldsymbol{y}^j(\tilde{k})\|_2)^2}{\sum_{i,j=1}^{m} d_{ij}(\tilde{k})^2}}, \tag{4.10}$$

where d_{ij} is the approximated geodesic distance between the snapshots $\boldsymbol{W}^i$ and $\boldsymbol{W}^j$ based on a k- or ε-neighborhood, $\boldsymbol{y}^i$ is the low-dimensional representative of the snapshots $\boldsymbol{W}^i$, $\|\boldsymbol{y}^i - \boldsymbol{y}^j\|_2$ the Euclidean distance between the corresponding representatives in the embedding space and $\tilde{k}$ stands for the number of nearest neighbors $k \in \mathbb{N} \setminus \{0\}$ or the radius $\varepsilon \in \mathbb{R}^+$ used to construct a k- or an ε-neighborhood, respectively. This function offers a direct comparison of the approximated geodesic distances and the corresponding distances in the embedding space. Another function that measures the quality of the embedding with respect to the neighborhood size is based on the *variance of distance ratios* [79, 78], i. e.

$$s(\tilde{k}) = \operatorname{var}\left[\left(\frac{\|\boldsymbol{y}^i(\tilde{k}) - \boldsymbol{y}^j(\tilde{k})\|_2}{d_{ij}(\tilde{k})} \right)_{i,j=1,\dots,m} \right], \tag{4.11}$$

where $\operatorname{var}[\cdot]$ denotes the variance. The smaller the variance is, the closer are the values of all ratios to the mean value and hence the better is the performance of the Isomap method. Verified by various experiments accomplished by Shi et al. [78], the parameter k^*, which yields the minimum of one of the functions above, i. e.

$$k^* = \arg\min_{\tilde{k}} s(\tilde{k}), \tag{4.12}$$

is a good parameter estimate for the size of the k- or ε-neighborhood of Isomap. Further techniques to estimate the neighborhood size are described in [77] and [71].

However, when an optimal embedding corresponding the loss function (4.4) is desired, the following appropriate stress function should be selected:

$$s(\tilde{k}) = \sum_{\substack{i,j=1 \\ i \neq j}}^{m} \frac{(\|\boldsymbol{y}^i(\tilde{k}) - \boldsymbol{y}^j(\tilde{k})\|_2 - d_{ij}(\tilde{k}))^2}{w_{ij}(\tilde{k})} \tag{4.13}$$

with w_{ij} as choosen for the loss function.

Due to the dependency of the actual number of neighbors, the stress functions (4.10), (4.11) and (4.13) show a step function behavior and, at least in the case of a k-neighborhood, the optimization problem (4.12) is an integer programming problem. Hence, ordinary optimizers cannot be applied, however, according to practical experiences, the stress functions are almost convex and hence the solution to equation (4.12) may be obtained by a method similar to interval section, which is detailed in Algorithm 4.3. As input, a list L of increasing neighborhood sizes has to be specified. Afterwards, if the list has more than two entries, the algorithm computes the stress for the $\lfloor \frac{1}{4}|L| \rfloor$-th, $\lfloor \frac{1}{2}|L| \rfloor$-th and $\lfloor \frac{3}{4}|L| \rfloor$-th entry of the list, which divides the list into four parts. Here, $\lfloor \cdot \rfloor$ denotes the floor function, i. e. $\lfloor x \rfloor = \max\{m \in \mathbb{Z} \mid m \leq x\}$. The two parts, which are not adjacent to the list value with the smallest stress, are removed. This step is repeated until one or two list elements are remaining. If the list consist of two elements, the corresponding stress values are compared to each other.

Algorithm 4.3 Computation of an optimal neighborhood size.

```
Require: List of increasing neighborhood sizes $L = [\tilde{k}_{min}, \dots, \hat{k}_{max}]$, stress function $s(\cdot)$
$L^* = s^* = \infty$
while $|L| > 1$ do
    for $i = \lfloor \frac{1}{4}|L| \rfloor, \lfloor \frac{1}{2}|L| \rfloor, \lfloor \frac{3}{4}|L| \rfloor$ do
        Compute embedding based on $L[i]$ neighbors
        if $s(L[i]) < s^*$ then
            $s^* \leftarrow s(L[i])$
            $L^* \leftarrow L[i]$
    if $\min\left(s(L[\lfloor \frac{1}{4}|L| \rfloor]), s(L[\lfloor \frac{1}{2}|L| \rfloor]), s(L[\lfloor \frac{3}{4}|L| \rfloor])\right) = s(L[\lfloor \frac{1}{4}|L| \rfloor])$ then
        delete entries $L[\lfloor \frac{1}{2}|L| \rfloor], \dots, L[|L|]$
    else if $\min\left(s(L[\lfloor \frac{1}{4}|L| \rfloor]), s(L[\lfloor \frac{1}{2}|L| \rfloor]), s(L[\lfloor \frac{3}{4}|L| \rfloor])\right) = s(L[\lfloor \frac{1}{2}|L| \rfloor])$ then
        delete entries $L[0], \dots, L[\lfloor \frac{1}{4}|L| \rfloor]$ and $L[\lfloor \frac{3}{4}|L| \rfloor], \dots, L[|L|]$
    else
        delete entries $L[0], \dots, L[\lfloor \frac{1}{2}|L| \rfloor]$
return $L^*$
```

4.4. From reduced-order representation to full-dimensional snapshot approximations

"The Isomap method only provides a mapping from the high-dimensional input space onto a lower-dimensional embedding space for a fixed finite set of given snapshots. For any ROM of the Navier-Stokes equations, however, it is an essential requirement that the approximate reduced-order flow solutions are of the same type and dimension as the full-order CFD snapshots. Hence, a back-mapping from the reduced-order embedding to the high-dimensional solution space is mandatory."(Franz et al. [36])

4.4.1. Non-parametric back-mapping

"The idea for the back-mapping is based on LLE [70] and was considered in similar form in [73]:

Suppose that $\{\boldsymbol{y}^1, \ldots, \boldsymbol{y}^m\} \subset \mathbb{R}^d$ is the low-dimensional representation of the full-order input snapshot set $\{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^m\} \subset \mathbb{R}^n$ embedded in $\mathbb{R}^d$ via Isomap.

For an arbitrary point $\boldsymbol{y}^* \in \mathbb{R}^d$ let $\mathcal{I}$, $|\mathcal{I}| = N$, denote the index set belonging to the N nearest neighbors of $\boldsymbol{y}^*$. Due to Isomap, the nearest neighbors $\{\boldsymbol{y}^j \mid j \in \mathcal{I}\}$ for a given vector $\boldsymbol{y}^* \in \mathbb{R}^d$ within the embedding space $\mathbb{R}^d$ correspond approximately isometrically to the nearest neighbors $\{\boldsymbol{W}^j \mid j \in \mathcal{I}\}$ on the full-order solution manifold. Therefore, a linear approximation of $\boldsymbol{y}^* \approx \sum_{j \in \mathcal{I}} a_j \boldsymbol{y}^j$ as a weighted sum of its N nearest neighbors is expected to yield good weighting coefficients that can be employed to build a linear combination of the corresponding high-dimensional snapshots on the full-order manifold.

The weights vector $\boldsymbol{a} = (a_{j_1}, \ldots, a_{j_N}) \in \mathbb{R}^N$, $j_i \in \mathcal{I}$, for a linear approximation of $\boldsymbol{y}^*$ by its neighbors is determined by the following optimality condition:

$$\min_{\boldsymbol{a} \in \mathbb{R}^N} \left\| \boldsymbol{y}^* - \sum_{j \in \mathcal{I}} a_j \boldsymbol{y}^j \right\|_2^2 \qquad \text{s. t.} \quad \sum_{j \in \mathcal{I}} a_j = 1. \tag{4.14}$$

This condition is motivated by the fact that Euclidean distances in the embedding space $\mathbb{R}^d$ approximately correspond to geodesic distances in the full-order solution manifold. The additional normalization constraint ensures that the reconstruction is invariant under linear translation. The translation invariance is needed for the final step (4.17) of the back-mapping.

When the number of neighbors in the neighborhood graph is larger than the dimensions of the embedding space, i.e., $N > d$, then the minimization problem (4.14) for finding the optimal weights is not uniquely solvable. This issue is addressed by Saul and Roweis [73] by adding a penalty term $\epsilon \|\boldsymbol{a}\|_2^2 = \epsilon \sum_{j \in \mathcal{I}} a_j^2$, $0 < \epsilon \ll 1$, to the cost function in (4.14), which regularizes the resulting system of equations and enforces the weights to be uniformly distributed in magnitude.

For our envisioned application of reduced-order modeling of aerodynamic problems, a more specific penalty term is needed, because even for a small number of nearest neighbors, some of them may still be too distant to contribute in a constructive way to the unknown $\boldsymbol{y}^*$. Furthermore, the reconstruction of a given snapshot $\boldsymbol{W}^i$ at $\boldsymbol{y}^* = \boldsymbol{y}^i$ by equation (4.14) is not guaranteed as the set of vectors $\{\boldsymbol{y}^j\}_{j \in \mathcal{I}}$ is linear dependent for $N > d$. Therefore, [a weighted norm on the weights is introduced] by setting

$$\|\boldsymbol{a}\|_c^2 := \sum_{j\in\mathcal{I}} c_j a_j^2, \quad c_j = \epsilon\left(\frac{\|\boldsymbol{y}^* - \boldsymbol{y}^j\|_2}{\max_i\{\|\boldsymbol{y}^* - \boldsymbol{y}^i\|_2\}}\right)^k, \quad 0 < \epsilon \ll 1,\ 1 < k \in \mathbb{N} \tag{4.15}$$

and the optimization problem (4.14) is modified accordingly:

$$\min_{\boldsymbol{a}} \left\|\boldsymbol{y}^* - \sum_{j\in\mathcal{I}} a_j \boldsymbol{y}^j\right\|_2^2 + \|\boldsymbol{a}\|_c^2 \qquad \text{s. t.} \quad \sum_{j\in\mathcal{I}} a_j = 1, \tag{4.16}$$

The penalty term limits the contribution of further away neighbors to be smaller than those of nearby neighbors. In order to accomplish this, the weights a_j are weighted by the distances to their corresponding neighbors. The distances are scaled to $(0, 1]$ by dividing the maximum distance $\|\boldsymbol{y}^* - \boldsymbol{y}^i\|_2$ to obtain similar magnitudes of distances for different neighborhoods. Afterwards, the scaled distances are taken to the power of k, achieving that the weights belonging to small values get even less restrictions than weights belonging to larger values. Finally, the weights are multiplied by a small value ϵ, which ensures that the influence of the penalty term on the optimization problem (4.14) remains small. Our best-practice values observed for these parameter are $k = 4$ and $\epsilon = 0.01$.

Now suppose that $\boldsymbol{a}^* \in \mathbb{R}^N$ is the solution to (4.16). Then the high-dimensional analogue $\boldsymbol{W}^* \in \mathbb{R}^n$ corresponding to $\boldsymbol{y}^*$ is

$$\boldsymbol{W}^* = \sum_{j\in\mathcal{I}} a_j^* \boldsymbol{W}^j. \tag{4.17}$$

In summary, the above steps define an approximate local inverse mapping from the low-dimensional embedding space to the full-order space, which will be referred to as the function $\boldsymbol{f}_{Rec}^N : \mathbb{R}^d \to \mathbb{R}^n$ in the following, where N denotes the number of nearest neighbors used for the local linear reconstruction."(Franz et al. [36])

Remark. "The additional penalty term induced by the weighted norm (4.15) reduces the sensitivity of the back-mapping to the number of nearest neighbors used for the local linear reconstruction."(Franz et al. [36]) By imposing larger restrictions to far-away points $\boldsymbol{y} \in Y$, the influence of the corresponding snapshots $\boldsymbol{W} \in W$ to the prediction $\boldsymbol{W}^*$ is getting less. Hence, assuming that the N^* nearest neighbors are much closer to $\boldsymbol{y}^*$ than the remaining points in Y, choosing a larger number of nearest neighbors $N > N^*$ does not have a significant effect on the prediction.

4.4.2. Solving the back-mapping weights optimization problem

"The optimal reconstruction weights given by (4.16) can be computed in closed form. It holds:

$$\begin{aligned}\Big\|\boldsymbol{y}^* - \sum_{j\in\mathcal{I}} a_j \boldsymbol{y}^j\Big\|_2^2 &= \Big\|\sum_{j\in\mathcal{I}} a_j(\boldsymbol{y}^* - \boldsymbol{y}^j)\Big\|_2^2 = \sum_{k=1}^{d}\Big(\sum_{j\in\mathcal{I}} a_j(y_k^* - y_k^j)\Big)^2 \\ &= \sum_{i,j\in\mathcal{I}}^{N} a_i a_j (\boldsymbol{y}^* - \boldsymbol{y}^i)^T(\boldsymbol{y}^* - \boldsymbol{y}^j) = \boldsymbol{a}^T G \boldsymbol{a},\end{aligned} \tag{4.18}$$

where G is the [positive semi-definite] *Gram matrix* with entries $G_{ij} = (\boldsymbol{y}^* - \boldsymbol{y}^i)^T(\boldsymbol{y}^* - \boldsymbol{y}^j)$. [Note that the first equality holds since $\sum_{j\in\mathcal{I}} a_j = 1$.] The cost function of the minimization problem (4.16) can now be rewritten as

$$\Big\|\boldsymbol{y}^* - \sum_{j\in\mathcal{I}} a_j \boldsymbol{y}^j\Big\|_2^2 + \sum_{j\in\mathcal{I}} c_j a_j^2 = \boldsymbol{a}^T G \boldsymbol{a} + \boldsymbol{a}^T C \boldsymbol{a} = \boldsymbol{a}^T \tilde{G} \boldsymbol{a}, \tag{4.19}$$

where $C = \operatorname{diag}(c_1, \dots, c_N)$ [is a diagonal matrix] and $\tilde{G} = G + C$ [is positive definite matrix]. Using a Lagrange ansatz forcing the weights to sum up to one, the reconstruction error (4.19) can be minimized analytically: Defining

$$\mathcal{L}(\boldsymbol{a}, \lambda) := \boldsymbol{a}^T \tilde{G} \boldsymbol{a} + \lambda(\boldsymbol{a}^T \mathbb{1} - 1) \tag{4.20}$$

it holds that

$$\nabla_{\boldsymbol{a}} \mathcal{L}(\boldsymbol{a}, \lambda) = 2\tilde{G}\boldsymbol{a} + \lambda \mathbb{1}, \quad \nabla_{\lambda} \mathcal{L}(\boldsymbol{a}, \lambda) = \mathbb{1}^T \boldsymbol{a} - 1, \tag{4.21}$$

where $\mathbb{1} = (1, \dots, 1) \in \mathbb{R}^N$. A necessary condition for an optimal solution is then given by the following system of linear equations:

$$\nabla \mathcal{L}(\boldsymbol{a}, \lambda) = 0 \Leftrightarrow \left\{\begin{array}{lcl} 2\tilde{G}\boldsymbol{a} + \lambda\mathbb{1} & = & 0 \\ \mathbb{1}^T \boldsymbol{a} - 1 & = & 0 \end{array}\right\} \Leftrightarrow \begin{pmatrix} 2\tilde{G} & \mathbb{1} \\ \mathbb{1}^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{a} \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{4.22}$$

Since the optimization problem is convex, as can be seen from (4.19), the unique zero of the gradient gives indeed the optimum."(Franz et al. [36])

4.4.3. Adaptive choice of the neighborhood size

The parameter N that specifies the size of the neighborhood employed to construct a full-order solution certainly affects the quality of the predictions. If insufficiently many neighbors are chosen, the information content will not suffice to construct an accurate full-order approximation. Conversely, if too many neighbors are employed, unwanted characteristics may appear. The latter effect is already damped by the weighted norm (4.15) introduced in Section 4.4.1. As an indicator of an insufficient neighborhood, i. e., a too small value of N, the magnitude of each component of the calculated weight vector $\boldsymbol{a} = (a_1, \ldots, a_N)$ can be exploited.

Let the vector $\boldsymbol{a}$ be given with $a_j \gg 1$ or $a_j \ll 0$ for $j \in \mathcal{J} \subsetneq \{1, \ldots, N\}$. Since it holds that $\sum_{i=1}^{N} a_i = 1$, there must be at least one weight a_l, $l \notin \mathcal{J}$, with $a_l \ll 0$ or $a_l \gg 1$, respectively. Without loss of generality, let $\mathcal{J} = \{j\}$ and hence $a_j \gg 1$ so that $a_l \ll 0$. Under this condition, the linear combination of full-oder snapshots (4.17) may yield an unphysical solution in the context of aerodynamics. Since the scaled unphysical snapshot $a_l \boldsymbol{W^l} = |a_l|(-\boldsymbol{W^l})$ is added to the corresponding upscaled snapshot $a_j \boldsymbol{W^j}$, a step-function behaviour of the shocks, bumps up- and downstream of the shocks or even unphysical values may appear. To avoid this, starting with a small neighborhood of size N, the number of nearest neighbors is successively increased by one until it holds that $\xi_l \leq a_i \leq \xi_u$ for all $i = 1, \ldots, N$ with a lower and upper bound of magnitude $\xi_l \in \mathbb{R}^-$ and $\xi_u \in \mathbb{R}^+$, respectively. After each increment of N, the weight vector $\boldsymbol{a}$ has to be recomputed by equation (4.22).

In practice, the bounds ξ_l and ξ_u are choosen from the intervals $[-0.25, -0.05]$ and $[1.05, 1.25]$, respectively, so that eventually $\boldsymbol{y}^*$ lies at least close to the convex hull of its N neighbors.

4.5. Isomap with interpolation

Similar to POD with interpolation of the POD expansion coefficients (see Section 3.3), a ROM based on Isomap combined with interpolation in the low-dimensional embedding space can now be build. "Let $W = \{\boldsymbol{W}^1, \ldots \boldsymbol{W}^m\} \subset \mathbb{R}^n$ be the sample set of linearly independent full-order CFD solutions, where each snapshot $\boldsymbol{W}^i$, depending on the parameter configuration $\boldsymbol{p}^i$, i. e., $\boldsymbol{W}^i = \boldsymbol{W}(\boldsymbol{p}^i)$, is a solution to the full-order system of ODEs (2.8). The procedure of computing reduced-order CFD approximations $\boldsymbol{W}^*$ at any untried parameter configuration $\boldsymbol{p}^* \in \mathcal{P}$ is as follows:

1. Apply Isomap to the data set W to obtain the corresponding low-dimensional embedding $Y = \{\boldsymbol{y}^1, \ldots, \boldsymbol{y}^m\} \subset \mathbb{R}^d$, where $\boldsymbol{y}^i$ is the low-dimensional representative of $\boldsymbol{W}^i$. Due to the relation between the snapshots $\boldsymbol{W}^i$ and their reduced-order representatives $\boldsymbol{y}^i$, it holds that at the sampled parameter combinations

$$\boldsymbol{W}(\boldsymbol{p}^i) = \boldsymbol{W}^i, \quad \Rightarrow \quad \boldsymbol{y}^i = \boldsymbol{y}(\boldsymbol{p}^i). \tag{4.23}$$

   Hence, the values of the unknown function $\boldsymbol{y} : \boldsymbol{p} \mapsto \boldsymbol{y}(\boldsymbol{p})$ are known at the sample points $\boldsymbol{p}^i$, $i = 1, \ldots, m$. [If the parametric dependency $\boldsymbol{p} \mapsto \boldsymbol{W}(\boldsymbol{p})$ is smooth and the true metric on the manifold along with the true smooth isometric embedding on the low-dimensional space is given, the smooth parametric dependency carries over to the embedding space $\boldsymbol{p} \mapsto \boldsymbol{y}(\boldsymbol{p})$. Under the assumption that Isomap gives a good approximation to the true, unknown smooth isometric embedding, the Isomap based embedding is treated in the same way.]

2. Based on this assumption, interpolation can be employed to obtain a new low-dimensional representative $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{p}^*) \in \mathbb{R}^d$ for any new parameter configuration $\boldsymbol{p}^*$ within the embedding space, where the sample data set is now taken to be $\{(\boldsymbol{p}^i, \boldsymbol{y}^i)\}_{i=1}^m$.

3. The reduced-order prediction of the corresponding CFD solution $\boldsymbol{W}^* = \boldsymbol{W}(\boldsymbol{p}^*)$ is constructed by applying the back-mapping introduced in Section 4.4 to the interpolated low-order representative $\boldsymbol{y}^*$ leading to

$$\boldsymbol{W}^* = \boldsymbol{f}_{Rec}^N(\boldsymbol{y}^*). \tag{4.24}$$

The above method will be referred to as *Isomap+I*."(Franz et al. [36])

Remark. "The most expensive calculations accrue in the process of setting up the ROM in the first step and are thus conducted offline and only once. Only the last two steps [of Isomap+I] are performed online and thus have to be executed for each new parameter configuration $\boldsymbol{p}^*$. The associated computational costs are detailed in [Section 6.2]."(Franz et al. [36])

4.6. Isomap with CFD-enhanced back-mapping

Given the same set-up as in Section 4.5, the flux residual is now exploited to obtain a prediction to the governing equations at an untried parameter configuration. For this purpose, a starting solution for the flux residual optimization may be obtained by equation (4.24) (see Remark in Section 3.4). Following the approach to minimize the CFD flux residual directly [95], i. e. solving

$$\min_{\boldsymbol{y} \in \mathbb{R}^d} \|\Omega^{-1} \mathbf{res}(\boldsymbol{W}(\boldsymbol{y}))\|_{L_2}^2, \tag{4.25}$$

the function $\mathbf{res}(\boldsymbol{W}(\boldsymbol{y}))$ depends on the representative $\boldsymbol{y} \in \mathbb{R}^d$, since

$$\boldsymbol{W}(\boldsymbol{y}) = \boldsymbol{f}_{Rec}^{N}(\boldsymbol{y}) = \sum_{j \in \mathcal{I}} a_j(\boldsymbol{y}) \boldsymbol{W}^j \tag{4.26}$$

holds. Again, $\mathcal{I}$ denotes the index set belonging to the N nearest neighbors of $\boldsymbol{y}$. Now, assuming that the interpolated representative $\boldsymbol{y}^*$ (see Section 4.5) is located in an appropriate neighborhood in the embedding space, $\boldsymbol{y}^*$ can be fixed so that the flow solution only depends on the weights $a_j, j \in \mathcal{I}$:

$$\boldsymbol{W}(\boldsymbol{a}) = \sum_{j \in \mathcal{I}} a_j \boldsymbol{W}^j, \quad \boldsymbol{a} = (a_{j_1}, \ldots, a_{j_N}),\ j_i \in \mathcal{I}. \tag{4.27}$$

Hence, the residual $\mathbf{res}(\boldsymbol{W}(\boldsymbol{a}))$ just depends on the coefficients $a_j, j \in \mathcal{I}$, and equation (4.25) becomes similar to equation (3.25). Because equation (4.25) is a nonlinear least-squares problem, the above method will be referred to as *Isomap+LSQ*.

In aerodynamic applications, parameters like the angle of attack or the Mach number determine the solution being sought after only at the farfield boundary. Since the number of boundary cells is negligible in comparision to the number of the remaining cells of a common CFD grid, a violation of the boundary conditions is almost unnoticeable by the objective function (4.25). Hence, the following constrained minimization problem is imposed:

$$\min_{\boldsymbol{a} \in \mathbb{R}^{|\mathcal{I}|}} \quad \|\Omega^{-1} \mathbf{res}(\boldsymbol{W}(\boldsymbol{a}))\|_{L_2}^2 \tag{4.28a}$$

$$\text{s. t.} \quad \|\mathbf{res}_b^I\| - \|\mathbf{res}_b(\boldsymbol{W}(\boldsymbol{a}))\| \geq 0, \tag{4.28b}$$

where $\mathbf{res}_b$ denotes the current LSQ based residual evaluated only at the boundary and, accordingly, $\mathbf{res}_b^I$ denotes the fixed initial residual at the boundary based on an Isomap+I prediction, which is employed as a initial solution for the minimization.

Contrary to POD with residual optimization reviewed in Section 3.4, where the *degrees of freedom (DoFs)* of the optimization problem depend on the number of modes k, the DoFs of Isomap with residual optimization is relative to the number of nearest neighbors N. Since it is expected that the Isomap based ROM selects the appropriate neighbors for a prediction and hence employs the proper local characteristics of the solution space, i. e., the relevant information content, it is expected that $N < k$ for similar accuracy of the optimized flow solutions. Furthermore, due to the smaller DoFs of the optimization problem, the least-squares problem should be numerically better-behaved and Isomap+LSQ should also outperform POD+LSQ concerning the CPU times.

Remark. The adaption of the constrained minimization problem (4.28) to POD+LSQ is straightforward.

4.7. Additional possible applications of Isomap(+I)

Isomap+I selects an appropriate set $\{W^j\}_{j\in\mathcal{I}}$ of $|\mathcal{I}| = N$ snapshots to predict a full-order solution for an unsampled parameter configuration $\boldsymbol{p}^*$. This neighborhood can also be exploited to build a local (reduced) POD of at most N modes and hence be utilized as a POD+I ROM or even employed in a Galerkin-POD framework to reduce the DoFs. Of course, obtaining a smaller number of DoFs has to be compared with the extra costs caused by building a small POD basis for each new prediction point $\boldsymbol{p}^*$ in the online phase.

A similar idea of local POD bases was followed before by Du and Gunzburger for projection-based ROMs [29]. Instead of taking advantage of the Isomap metric, a centroidal Voronoi tessellation (CVT) based on a distance function, which measures the distance from an one-dimensional subspace to a q-dimensional subspace, is performed in the high-dimensional solution space to construct snapshot clusters. For each cluster, a POD is performed and the solution of the governing system is replaced by the agglomeration of all POD modes. Later, this concept was improved by Amsallem et al. [4]. In place of using the entire linear combination of POD modes, the snapshot cluster centroid closest to the current state is determined at each time-step and the state update is approximated using only the corresponding POD basis, or, more generally, the corresponding reduced-order basis (ROB). The norm for the distance computations is chosen to be $\|\boldsymbol{W}\|_H = \sqrt{\boldsymbol{W}^T H \boldsymbol{W}}$, where $\boldsymbol{W} \in \mathbb{R}^n$ and $H \in \mathbb{R}^{n\times n}$ is a symmetric positive definite or semi-definite matrix defining a metric or pseudo-metric in $\mathbb{R}^n$, respectively. Furthermore, neighboring snapshots are added to each cluster to obtain an overlapping clustering, which can be expected to improve the reduced-order solution when the nearest centroid to the state trajectory is changing.

This idea can also be adapted to parametric ROMs based on Isomap: First, a CVT based on Euclidean distances may be performed in the embedding space obtained by Isomap to build overlapping clusters while exploiting the Isomap metric. Afterwards, POD is applied to each cluster to move the costs caused by building small POD bases to the offline phase. The POD basis belonging to the centroid closest to $\boldsymbol{y}^*$, which is obtained as detailed in step 2 in Section 4.5, is employed for a prediction at $\boldsymbol{p}^*$, accomplished as stated in Section 3.3.

CHAPTER 5

Adaptive sampling

The approximation of the manifold and hence the quality of the ROM, i. e., the algorithmic efficiency and numerical accuracy, strongly depends on the number of sampled parameters and their location in the parameter space. It is intuitively obvious that the higher the dimension of the parameter space, the more snapshots are needed to build an accurate ROM: "if a certain level of prediction accuracy is achieved by sampling a one-variable space in n locations, to achieve the same sample density in a k-dimensional space, n^k observations are required."[33, Section 1.1] The latter fact is often referred to as the *curse of dimensionality*. To keep the number of calculations of full-order snapshots small, an incremental sampling method will be developed, that attempts to create a homogeneously distributed data set of the manifold to reduce the lack of information as much as possible.

In contrast to the Greedy algorithm for stationary and POD-Greedy algorithm for time-dependent RB approaches introduced in [89] and [42], respectively, the basic version of the method developed here does not depend on a-posteriori error bounds but on the geometrical characterization of the manifold obtained by Isomap. Thus, no expensive evaluations of the residual at a large set of test parameters are necessary, which makes the proposed method similar to sample infill criteria strategies like mean squared error (MSE) based exploration or expected improvement (cf. [33, Section 3.2]). The latter methods are often used in terms of surrogate based optimization.

The adaptive sampling strategy based on the manifold characterization is developed in Section 5.1, followed by a proof of concept in Section 5.2.

5.1. Manifold filling adaptive sampling strategy

As already noted in Section 4.2, ML methods preserve the geometry of the underlying manifold $\mathcal{W}$ in a certain manner. In case of Isomap, an approximately isometric embedding Y of the data manifold $W \subset \mathcal{W}$ is obtained (see Section 4.3). Considering the isometry between the manifold and the embedding space, the following fact can be exploited for an adaptive sampling strategy:

> Let the representatives $\boldsymbol{y}^1, \dots, \boldsymbol{y}^m \in Y \subset \mathbb{R}^d$ be given by the map $\boldsymbol{y} : \mathcal{P} \to \mathbb{R}^d$ with $\boldsymbol{y}(\boldsymbol{p}^j) = \boldsymbol{y}^j$, $\boldsymbol{p}^j \in \mathcal{P} \subset \mathbb{R}^d$, $j = 1, \dots, m$ (see Section 4.5). If there exists an δ-ball $B_\delta(\boldsymbol{y}^*) = \{\tilde{\boldsymbol{y}} \in \mathbb{R}^d \mid \|\boldsymbol{y}^* - \tilde{\boldsymbol{y}}\|_2 < \delta\}$ with $\delta \gg 0$ and $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{p}^*) \in \mathbb{R}^d$, $\boldsymbol{p}^* \in \mathcal{P} \subset \mathbb{R}^d$, which does not contain any representative $\boldsymbol{y}^j$, i. e., $\forall \tilde{\boldsymbol{y}} \in Y : \tilde{\boldsymbol{y}} \notin B_\delta(\boldsymbol{y}^*)$, then there is a lack of information in the corresponding region $\boldsymbol{f}_{Rec}(\{\boldsymbol{y}(\boldsymbol{p}) \in B_\delta(\boldsymbol{y}^*) \mid \boldsymbol{p} \in \mathcal{P}\})$, $\boldsymbol{f}_{Rec} : \{\boldsymbol{y}(\boldsymbol{p}) \in \mathbb{R}^d \mid \boldsymbol{p} \in \mathcal{P}\} \to \mathcal{W}$, on the manifold $\mathcal{W}$ specified by the data set W.

Filling these *"holes"* in the manifold by adding the solution $\boldsymbol{W}(\boldsymbol{p}^*)$, $\boldsymbol{p}^* \in \mathcal{P}$, to the data manifold should lead to a more evenly distributed sampling of the data manifold W, hence to a better characterization of the manifold $\mathcal{W}$ and eventually to a more accurate ROM.

Let the parameter space be given by the bounded set $\mathcal{P} \subset \mathbb{R}^d$ and let the discrete subset $P_{\tilde{m}} = \{\boldsymbol{p}^1, \dots, \boldsymbol{p}^{\tilde{m}}\} \subset \mathcal{P}$ be a set of $\tilde{m} \in \mathbb{N}$ preselected parameter configurations. Moreover, let $1 \le i \le m - \tilde{m}$ be the number of the current iteration of the adaptive sampling process, where $i, m \in \mathbb{N}$ and $m > \tilde{m}$ is the number of desired snapshots. Starting with the initial sampling of $\tilde{m}$ snapshots $W_{\tilde{m}} = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^{\tilde{m}}\} \subset \mathbb{R}^n$, where $\boldsymbol{W}^j$ is the solution to the parameter configuration $\boldsymbol{p}^j \in \mathcal{P}$, the initial embedding $Y_{\tilde{m}} = \{\boldsymbol{y}^1, \dots, \boldsymbol{y}^{\tilde{m}}\} \subset \mathbb{R}^d$ is calculated via Isomap. To identify holes in the data manifold of the current sampling, regions of low density within the embedding space have to be detected. For a given point $\boldsymbol{p} \in \mathcal{P}$ the *weighted sum of the distances of the* $N \in \mathbb{N}$ *nearest neighbors* to $\boldsymbol{y}(\boldsymbol{p})$ is calculated:

$$dist(\boldsymbol{y}(\boldsymbol{p})) := \frac{d_{min}(\boldsymbol{y}(\boldsymbol{p}))}{d_{max}(\boldsymbol{y}(\boldsymbol{p}))} \sum_{j \in \mathcal{I}} \|\boldsymbol{y}(\boldsymbol{p}) - \boldsymbol{y}^j\|_2, \tag{5.1}$$

where $\mathcal{I}$, $|\mathcal{I}| = N$, denotes the index set belonging to the N nearest neighbors of $\boldsymbol{y}(\boldsymbol{p})$, $d_{min}(\boldsymbol{y}(\boldsymbol{p})) = \min_{j \in \mathcal{I}} \|\boldsymbol{y}(\boldsymbol{p}) - \boldsymbol{y}^j\|_2$ and $d_{max}(\boldsymbol{y}(\boldsymbol{p})) = \max_{j \in \mathcal{I}} \|\boldsymbol{y}(\boldsymbol{p}) - \boldsymbol{y}^j\|_2$. To ensure that $\boldsymbol{p} \in \mathcal{P}$ is fulfilled during optimization, function (5.1) is multiplied by an indicator function ω:

$$E_{dist}(\boldsymbol{y}(\boldsymbol{p})) := dist(\boldsymbol{y}(\boldsymbol{p})) \cdot \omega(\boldsymbol{p}), \tag{5.2}$$

where

$$\omega(\boldsymbol{p}) = \begin{cases} 1 & \text{if } \boldsymbol{p} \in \mathcal{P}, \\ 0 & \text{else.} \end{cases} \tag{5.3}$$

The point $\boldsymbol{p}^* \in \mathcal{P}$, which maximizes the sum of distances, i. e. $\boldsymbol{p}^* = \arg\max E_{dist}(\boldsymbol{y}(\boldsymbol{p}))$, is assumed to be the searched point. The above method will be referred to as maximum distance error (MDE).

The error plot of an initial embedding (see Section 5.2) corresponding to E_{dist} is depicted in Figure 5.1(a). Due to the function ω given by equation (5.3), the error in Figure 5.1(a) vanishes as soon as the corresponding parameter lies outside the parameter space $\mathcal{P}$, even if the distances measured in the embedding space are still increasing (see Figure 5.1(b)). This effect is represented by the blue area outside of the scattered data. Furthermore, the error function is designed to have zero error at each parameter combination $\boldsymbol{p} \in P_l, l \in \{\tilde{m}, \tilde{m}+1, \dots, m\}$.

To detail the sampling process, let $Y_{\tilde{m}+i-1} = \{\boldsymbol{y}^1, \dots, \boldsymbol{y}^{\tilde{m}+i-1}\}$ be the embedding obtained by applying Isomap to the data manifold $W_{\tilde{m}+i-1} = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^{\tilde{m}+i-1}\}$. Let $\boldsymbol{p}^* \in \mathcal{P}$ be the ith point obtained by MDE during the ith iteration of the adaptive sampling, whereby the unknown function $\boldsymbol{y} : \mathcal{P} \to \mathbb{R}^d$ is evaluated via interpolation based on the data set $\{(\boldsymbol{p}^j, \boldsymbol{y}^j)\}_{j=1}^{\tilde{m}+i-1}$ (cf. Section 4.5, step 2). Then, a CFD simulation at the parameter configuration $\boldsymbol{p}^* \in \mathcal{P}$ is conducted to achieve a full-order solution $\boldsymbol{W}^* = \boldsymbol{W}(\boldsymbol{p}^*)$, which is assumed to improve the description of the manifold $\mathcal{W}$ by the data manifold W. Just after adding the recently calculated snapshot $\boldsymbol{W}^*$ to the data set $W_{\tilde{m}+i-1}$, the next iteration based on the data set $W_{\tilde{m}+i} = \{\boldsymbol{W}^1, \dots, \boldsymbol{W}^{\tilde{m}+i-1}, \boldsymbol{W}^{\tilde{m}+i} := \boldsymbol{W}^*\}$ is performed. A pseudo code of this method is outlined in Algorithm 5.1.

Assuming that the snapshots $\boldsymbol{W}^1, \dots, \boldsymbol{W}^{\tilde{m}+i-1}$ corresponding to the current set of parameters $\{\boldsymbol{p}^1, \dots, \boldsymbol{p}^{\tilde{m}+i-1}\}$ yield a homogeneously sampled manifold, a reconstruction error based on the N nearest neighbors may be introduced. For each $\boldsymbol{y}^j \in Y_{\tilde{m}+i-1}$, a prediction $\hat{\boldsymbol{W}}(\boldsymbol{y}^j) = \hat{\boldsymbol{W}}^j$ based on its N nearest neighbors (cf. Section 4.4) is computed and the relative error $E_{rel}(\boldsymbol{y}^j) = E_{rel}^j = \frac{\|\hat{\boldsymbol{W}}^j - \boldsymbol{W}^j\|_2}{\|\boldsymbol{W}^j\|_2}$ to the corresponding snapshot $\boldsymbol{W}^j$ is calculated. Note that $\boldsymbol{y}^j$ is not counted as a neighbor of itself and hence $\hat{\boldsymbol{W}}^j \neq \boldsymbol{W}^j$. Afterwards, interpolation is exploited to approximate the relative error at $\boldsymbol{y} \notin Y_{\tilde{m}+i-1}$ based on the data set $\{(\boldsymbol{y}^j, E_{rel}^j)\}_{j=1}^{\tilde{m}+i-1}$. To ensure that the error is zero at given sample points, the reconstruction error is defined as

$$E_{rec}(\boldsymbol{y}(\boldsymbol{p})) := E_{rel}(\boldsymbol{y}(\boldsymbol{p})) \cdot E_{dist}(\boldsymbol{y}(\boldsymbol{p})). \tag{5.4}$$

A homogeneously sampled manifold is a crucial assumption for the application of the error E_{rec} as otherwise there would be no reliable information of the actual reconstruction error at some part of the manifold. For example, let the sampling consists of snapshots which are separated in two dense clusters in the embedding space. In this case, the data manifold is not homogeneously sampled, since there is a gap between the two clusters. Applying the error E_{rec} to this sampling, the interpolation of the error within the gap would not yield a reliable value as there is no actual information on the reconstruction error.

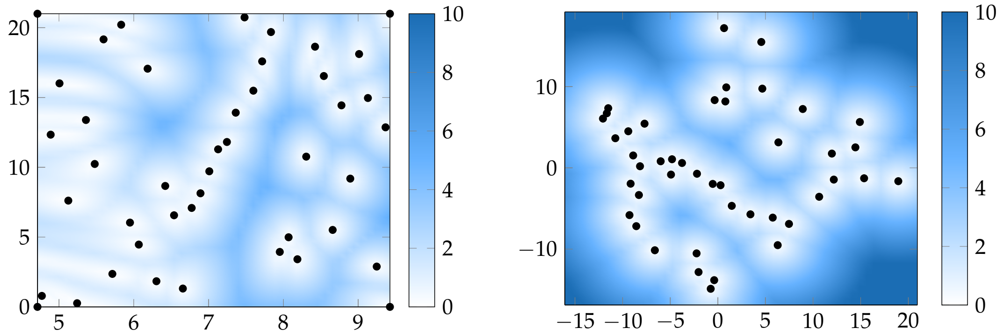
The corresponding error plot is shown in Figure 5.1(d). Compared to E_{dist}, E_{rec} yields a better match to the actual relative error depicted in Figure 5.1(e). Since an almost homogeneously sampled manifold must be given, the error function E_{rec} should only be employed every k iterations in practice. For the remaining iterations E_{dist} is utilized to ensure a homogeneously distributed manifold. The hybrid sampling strategy, maximizing E_{rec} at every kth iteration and E_{dist} otherwise, is referred to as hybrid error (HYE) in the following.

Remark. Due to the sequentially collected snapshots, a full Isomap procedure has to be performed after each iteration. A remedy may be given by an incremental version of Isomap, which yields an accurate low-dimensional representation of the data in an efficient manner [52]. However, compared to a full-order CFD solution also computed after each iteration, the computational costs of performing Isomap may be neglected.

Choice of the initial sampling and starting points for optimization. The adaptive method depends on the choice of the initial sampling and also on good starting points for the maximization as the objective function features plenty local maxima. If the boundaries of the parameter space $\mathcal{P}$ are specified by the initial sampling, the convex hull of Y should also determine the boundaries within the embedding space. Hence, performing a *Delaunay triangulation* of Y (see Appendix B), the parameters $\boldsymbol{p}^i = \boldsymbol{p}(\boldsymbol{c}^i)$, $i = 1, \dots, l$, belonging to the centers $\boldsymbol{c}^i \in \operatorname{conv}(Y)$, $i = 1, \dots, l$, of the simplices with $l \in \mathbb{N}$ largest volumes yield good starting points for the optimization of E_{dist}. The unknown functional coherence $\boldsymbol{p} : \operatorname{conv}(Y) \to \mathbb{R}^d$ is computed via interpolation. If the boundaries of the parameter space $\mathcal{P}$ are not included, the convex hull of Y only describes a part of the manifold $\mathcal{W}$, hence the starting points are restricted to a subset $\tilde{\mathcal{P}} \subset \mathcal{P}$. In that case, remedy is given by performing space filling random samplings like Latin hypercube sampling (LHS) [33] as starting points in the first few iterations to extend the description of the manifold. The latter is also suggested as starting points for the optimization of E_{rec} to increase the probability to locate the global maximum, since the error function does not only depend on distances any more.

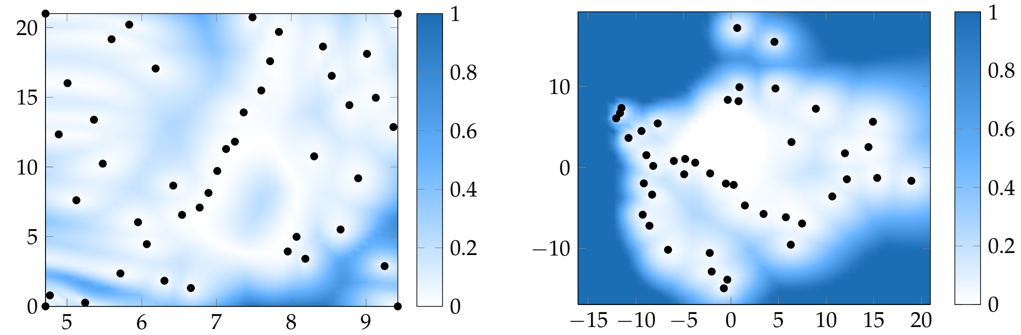
5.2. Proof of concept

Before the proposed Algorithm 5.1 is fully executed, a single iteration of the sampling strategy is performed to validate MDE for determining the holes. Thereto, the *swiss roll* including a hole, called *swiss hole* in the literature, is given (see Figure 5.2(c)). The



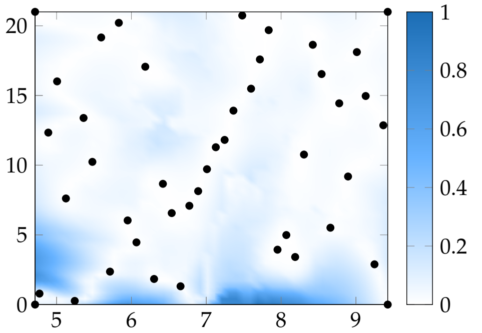
(a) E_{dist} error using only the nearest neighbor ($N = 1$).

(b) Distances using only the nearest neighbor ($N = 1$) measured in the embedding space.



(c) E_{rec} error using only the nearest neighbor ($N = 1$).

(d) Distances using only the nearest neighbor ($N = 1$) measured in the embedding space and multiplied by the relative error E_{rel}.



(e) Actual relative error measured in 2-norm and computed for each parameter combination given by a 25×25 grid.

Figure 5.1. Error plots of MDE and HYE. In Figures (a) and (c), the error E_{dist} and E_{rec} are shown in the parameter space, respectively. Figures (b) and (d) illustrate the corresponding error surfaces in the embedding space. The actual relative error is depicted in Figure (e).

Algorithm 5.1 Manifold filling adaptive sampling algorithm.

```
Require: Desired number of snapshots $m$, number of initial snapshots $\tilde{m}$
Generate $\tilde{m} < m$ parameter values $\boldsymbol{p}^1, \dots, \boldsymbol{p}^{\tilde{m}} \in \mathcal{P}$, e. g. via LHS
$P \leftarrow \{\boldsymbol{p}^1, \dots, \boldsymbol{p}^{\tilde{m}}\}$
Compute full-order CFD solutions $\boldsymbol{W}(\boldsymbol{p})$ at each parameter value $\boldsymbol{p} \in P$
$W \leftarrow \{\boldsymbol{W}(\boldsymbol{p}^1), \dots, \boldsymbol{W}(\boldsymbol{p}^{\tilde{m}})\}$        ▷ initial sampling
for $i = 1$ to $m - \tilde{m}$ do
    Calculate embedding $Y$ of the generated snapshot set $W$ via Isomap
    Compute interpolation model for $\boldsymbol{y}$ based on $\{(\boldsymbol{p}^j, \boldsymbol{y}^j)\}_{j=1}^{\tilde{m}+i-1}$
    Determine $\boldsymbol{p}^* \in \mathcal{P}$ by maximizing $E_{dist}$ or $E_{rec}$
    Compute full-order CFD solution $\boldsymbol{W}^*$ at parameter configuration $\boldsymbol{p}^* \in \mathcal{P}$
    $P \leftarrow P \cup \{\boldsymbol{p}^*\}$
    $W \leftarrow W \cup \{\boldsymbol{W}^*\}$
return Set $W$ of $m$ snapshots
```

swiss roll itself is parameterized by two parameters t and h:

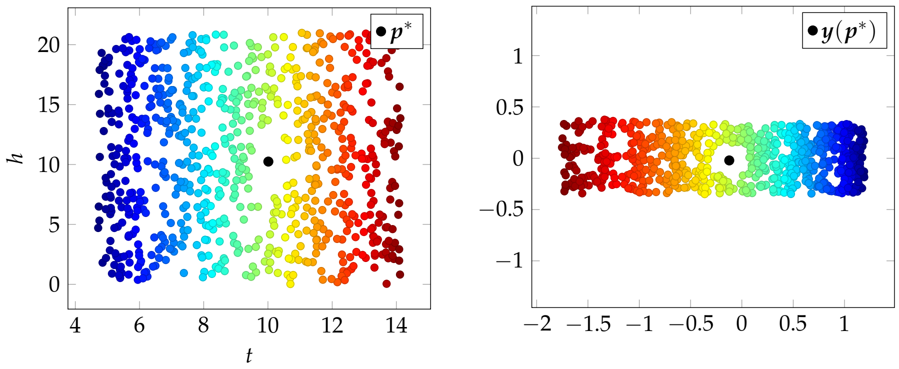
$$\begin{aligned} \boldsymbol{s} &: \mathcal{P} \to \mathbb{R}^3, \\ (t, h) &\mapsto (t\cos(t), h, t\sin(t)), \end{aligned}$$

where $\mathcal{P} = [\frac{3}{2}\pi, \frac{9}{2}\pi) \times [0, 21)$. The hole is accomplished by restricting the parameter space to $\mathcal{P} \setminus (9.5, 10.5) \times (8, 13)$ (see Figure 5.2(a)). The corresponding Isomap-embedding of the swiss hole given by a discrete subset $P \subset \mathcal{P}$, $|P| = 748$, is depicted in Figure 5.2(b). The goal is to determine a point $\boldsymbol{p}^* \in \mathcal{P}$, so that $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{p}^*) \in \mathbb{R}^2$ lies within the hole of the embedding space between the yellow and green colored points (see Figure 5.2(b)). Afterwards, the corresponding full-order solution $\boldsymbol{s}(\boldsymbol{p}^*) \in \mathbb{R}^3$ is calculated, which is assumed to lie within the hole of the manifold in Figure 5.2(c). The black dot depicted in Figure 5.2(b) is the interpolated representative $\boldsymbol{y}(\boldsymbol{p}^*)$ of the optimal solution $\boldsymbol{p}^*$ computed by the MDE strategy. Here, E_{dist} is calculated by using the $k = 1$ nearest neighbors. As it can be seen, $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{p}^*)$ lies within the hole in the embedding space and, furthermore, the full-order solution $\boldsymbol{s}(\boldsymbol{p}^*) \in \mathbb{R}^3$ also fills the hole in the manifold (see Figure 5.2(c)).

Now, a full sampling process of a curved plane in $\mathbb{R}^3$ is performed. An analytical description of the plane is given by

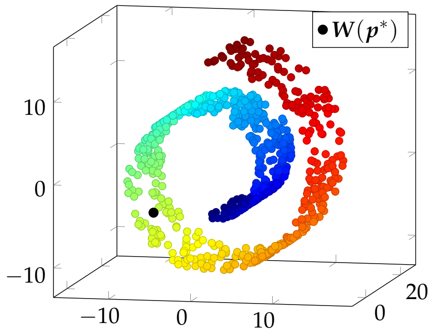
$$\begin{aligned} \boldsymbol{c} &: \mathcal{P} \to \mathbb{R}^3, \\ (t, h) &\mapsto (\tfrac{t^2}{10}\cos(t), h, \tfrac{t^2}{10}\sin(t)), \end{aligned}$$

where the parameter space is restricted to by $\mathcal{P} = [\frac{3}{2}\pi, 3\pi] \times [0, 21]$. The parameter



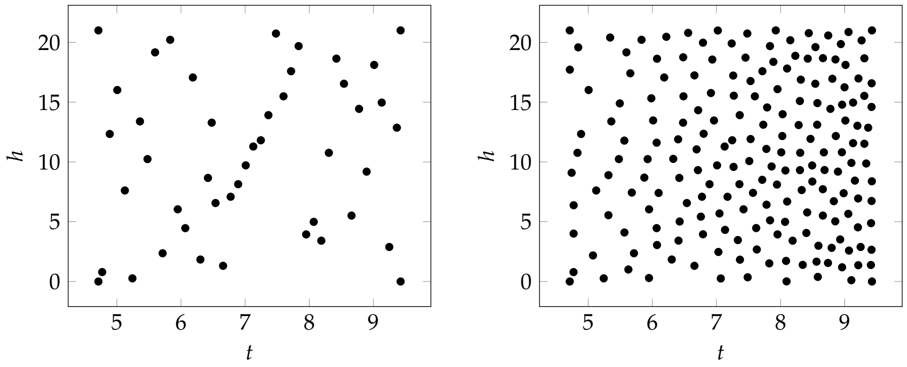
(a) Sampled parameters within the parameter space $\mathcal{P} \setminus (9.5, 10.5) \times (8, 13)$.

(b) Embedding of the swiss hole. The corresponding point within the hole determined by maximizing E_{dist} is depicted by the black dot.



(c) Swiss roll with a hole. The newly calculated high dimensional solution based on the determined point within the embedding space is depicted by the black dot.

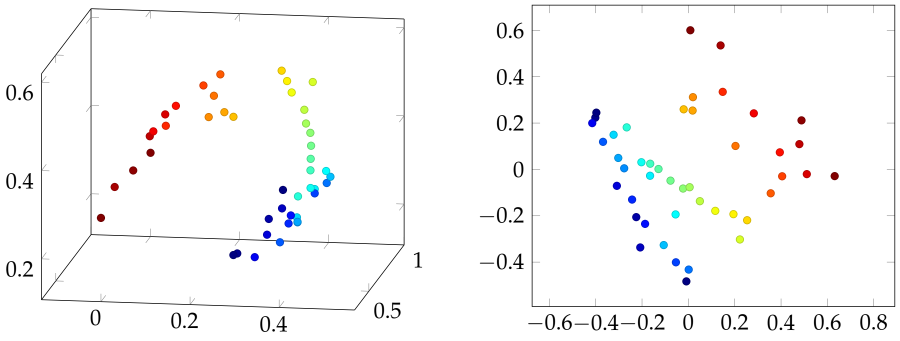
Figure 5.2. Detection of a hole within a manifold illustrated for the swiss hole.



(a) Initial parameter sampling (LHS). **(b)** Parameter sampling after conducting MDE.

Figure 5.3. Locations of the parameter samples.

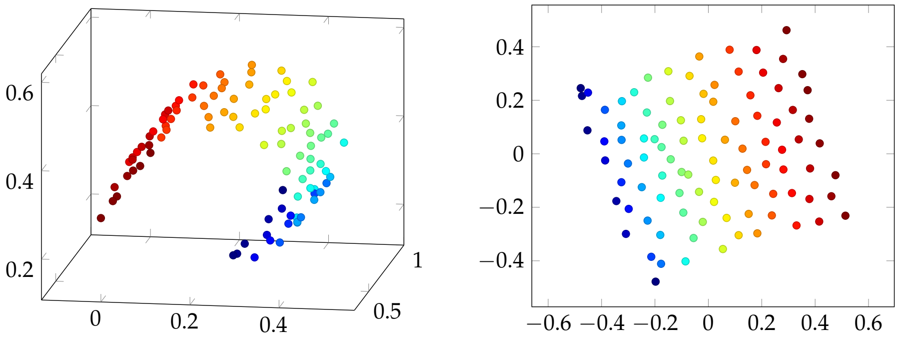
locations are generated by a full factorial design at the boundaries of $\mathcal{P}$ (edges only) and a Latin hypercube sampling within $\mathcal{P}$ (cf. Figure 5.3(a)). The objective is to generate a homogenous description of the curved plane in a three-dimensional space based on an initial sampling W_{44} of 44 snapshots, which is depicted in Figure 5.4(a). Figure 5.4(b) shows the initial embedding. Note that due to the growing angular velocity of the function $\boldsymbol{c}$ for increasing t, the final parameter sampling should be denser for larger values of t to obtain an evenly distributed sampling of the curved plane. The MDE method was chosen to detect the regions of low sampling density. As initial guesses for the optimization procedures, a LHS of size 30 has been performed for each iteration. In Figure 5.5, the generated snapshot sets W_m, $m \in \{94, 144, 194\}$, and their embeddings after 50, 100 and 150 iterations are shown, respectively. The number of nearest neighbors used for the embedding was chosen automatically in each iteration (see Section 4.3.3). As can be seen from Figure 5.5, the more iterations were performed the more evenly is the sampling of the data manifold and, due to the idea of the method, the embedding as well. The 194 parameter locations of the final snapshot set are depicted in Figure 5.3(b). As expected, the sampling is denser for larger t, which is due to the higher angular velocity with respect to t.



(a) Data manifold based on 44 samples shown in the high dimensional solution space.

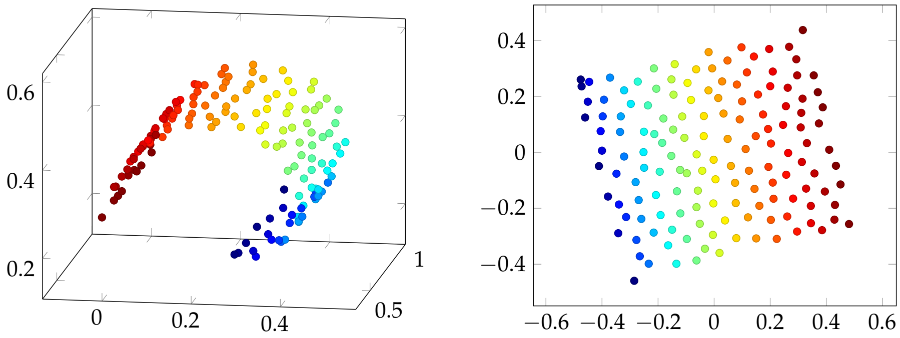
(b) Corresponding embedding space.

Figure 5.4. Initial data manifold and its embedding.



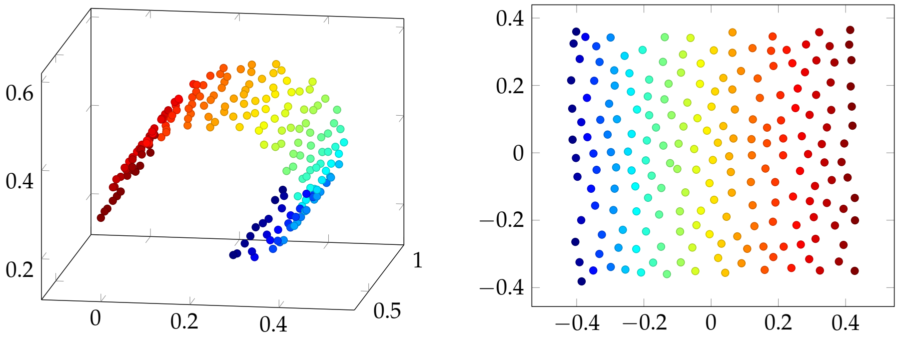
(a) Data manifold based on 94 samples shown in the high dimensional solution space.

(b) Corresponding embedding space.



(c) Data manifold based on 144 samples shown in the high dimensional solution space.

(d) Corresponding embedding space.



(e) Data manifold based on 194 samples shown in the high dimensional solution space.

(f) Corresponding embedding space.

Figure 5.5. Manifold filling adaptive sampling strategy illustrated for a curved plane.

CHAPTER 6

Computational issues

6.1. Software integration

For an efficient implementation of the developed methods, *SMARTy*, a toolbox for surrogate modeling for aero-data programmed in Python[1], is our framework of choice. The user friendly application programming interface (API) along with features such as an interpolation, an optimization and a snapshot module was developed at the DLR by Niklas Karcher[2]. Since SMARTy provides an interface to the DLR TAU code and to the FlowSimulator DataManager (FSDM)[3], which provides a broad library of classes for in-memory storage and parallel management of data associated with CFD, the design of experiment (DoE) and the adaptive sampling can also be conducted within the SMARTy framework. An overview of the major modules and classes of the SMARTy toolbox is given in Figure 6.1.

Due to the modular and flexible environment of SMARTy, the implementation of the class `Isomap` with an interface similar to the class `POD`, which provides the POD detailed in Chapter 3, was possible. The class `Isomap` features the ML method Isomap (see Section 4.3) along with its back-mapping introduced in Section 4.4. Within this class the fundamental package for scientific computing with Python, namely *NumPy*[4], and various numerical routines of the *SciPy* library[5] are exploited. Additionally, a common interface for an adaptive sampling module was developed to join parameter space and embedding space-based sampling strategies.

---

[1] programming language, https://www.python.org/

[2] niklas.karcher@dlr.de

[3] http://software.dlr.de/p/fsdm/home/

[4] http://www.numpy.org

[5] http://www.scipy.org/scipylib/index.html

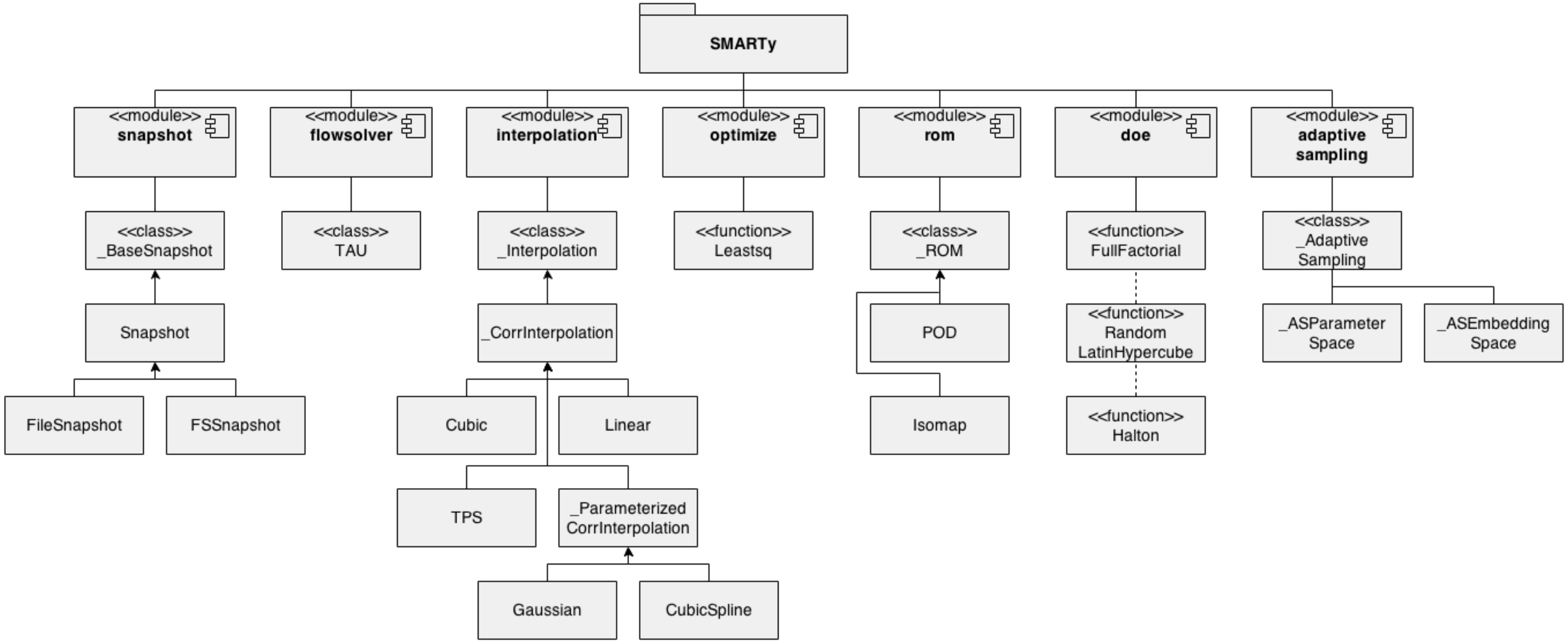


Figure 6.1. Major modules and classes of the SMARTy toolbox.

6.2. Complexity of Isomap+I

"Let m be the number of snapshots and let n, d denote the dimensions of the full-order and the embedding space, respectively. Moreover, let N be the number of nearest neighbors used for building the Isomap neighborhood graph and let $|\mathrm{E}|$ be the number of edges within this graph. Recall that the N_{rec} nearest neighbors are used for the back-mapping."(Franz et al. [36])

The first step of the Isomap+I algorithm is to construct an undirected weighted neighborhood graph $G = (V, E, c)$ (see Seciton 4.3.1). For sequential computations a kd-tree [15] is constructed in $\mathcal{O}(nm \log m)$ FLOPS [57] and employed to find the N nearest neighbors. "Even though the depth of the resulting tree may theoretically exceed $\mathcal{O}(\log m)$, it is only of logarithmic depth in generic practical examples [57]. As can be seen from Algorithm 4.1, the computational complexity for approximating the geodesic distances is of order $\mathcal{O}(m \cdot (N \log m + N) + m^2 + n|\mathrm{E}| + m^3)$, since for each of the m given snapshots $\boldsymbol{W}^i$, the N nearest neighbors $\boldsymbol{W}^{j_i}$ must be determined, which takes $\mathcal{O}(N \log m)$ FLOPS each, and the corresponding edges must be added to the neighborhood graph. The remaining terms stem from applying the Floyd-Warshall algorithm together with the initialization of the distance matrix, which take $\mathcal{O}(m^2 + n|\mathrm{E}| + m^3)$. The [classical] d-dimensional embedding determined by MDS is given by the eigenvectors corresponding to the d largest eigenvalues of the $m \times m$ double-centered distance matrix. For a dense eigenvalue solver, the cost is approximately $\mathcal{O}(dm^2)$."(Franz et al. [36]) Applying variable alternation as described in Algorithm 4.2 additional costs of $\mathcal{O}(ldm^2)$ accrue, where l is the maximum number of iterations of the outer loop. Furthermore, if the proper neighborhood size for the neighborhood graph is automatically determined (see Section 4.3.3), the former costs arise $\mathcal{O}(m)$ times. "The correlation matrix of a radial basis functions (RBF) interpolation model [46] is build in $\mathcal{O}(dm^2)$ FLOPS. Directly solving the resulting linear system of equations to determine the weights of the interpolation model is done in $\mathcal{O}(m^3)$ FLOPS. The precise interpolation scheme may be replaced subject to the user's preference, of course. For a more efficient online prediction, a kd-tree is also constructed for the embedded reduced-order representation, which is calculated in $\mathcal{O}(dm \log m)$."(Franz et al. [36])

In summary, the overall complexity of the offline stage without any CFD computations of the proposed Isomap+I method is

$$\begin{aligned}\mathcal{O}(nm \log m + m \cdot (N \log m + N) + m^2 + n|\mathrm{E}| + m^3 \\ + dm^2 + m^3 + dm \log m) = \mathcal{O}(n(m \log m + |E|))\end{aligned} \quad (6.1)$$

for the classical solution and

$$\mathcal{O}((nm\log m + m\cdot(N\log m + N) + m^2 + n|\mathrm{E}| \\ + m^3 + dm^2 + m^3 + dm\log m + ldm^2)\cdot m) = \mathcal{O}(nm(m\log m + |E|)) \quad (6.2)$$

otherwise, where $d, |E|, N, m, l \ll n$ holds.

"In the online stage, only $\mathcal{O}(dm + N_{rec}\log m + N_{rec}^3 + N_{rec}n)$ FLOPS accrue to predict a full-order solution. This is due to the fact that a new representative $\boldsymbol{y}^*$ is estimated via RBF interpolation in $\mathcal{O}(dm)$ FLOPS and, after identifying the N_{rec} nearest neighbors of $\boldsymbol{y}^*$, a linear system of dimension $(N_{rec}+1)$ for the back-mapping weights problem has to be solved. The only operation that (necessarily) scales in the full-order dimension n is the back-mapping to the full-order space via (4.17), which takes $\mathcal{O}(N_{rec}n)$ FLOPS.

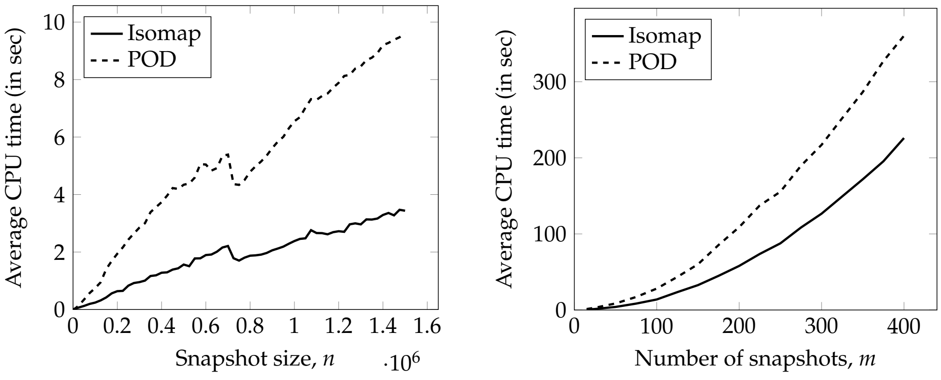
In standard ROM applications, the full-order dimension n exceeds by far all other quantities, i.e., $n \gg m, d, |\mathrm{E}|, N, N_{rec}$. In this regard, it is [most] important to note that both the online and the offline stage of the Isomap+I ROM scale *linearly* in n. Online computation of the reduced-order coordinates is actually independent of n and thus qualifies as a real-time method.

Figures 6.2(a) and 6.2(b) show the average CPU time of the offline stage (corresponding to setting up the ROM) depending on the dimension of the full-order space, n, and on the number of snapshots, m, respectively."(Franz et al. [36]) Here, the number of nearest neighbors employed by Isomap to identify the data manifold is set to 10 and a two dimensional embedding by doing a full eigenvalue decomposition, keeping only the eigenvectors corresponding to the two largest eigenvalues, is calculated. Additional variable alternation is not applied. The results for Isomap are compared to those of POD. As can be seen, in both cases the runtime of Isomap behaves very similar to that of POD, however, Isomap clearly outperforms POD in terms of absolute runtime.

6.3. Notes on parallelization

Applying the TAU solver to seek for accurate CFD solutions implies the usage of a dense grid for the underlying aircraft or wing. The number of grid points n_g may exceed $10\cdot 10^6$, and hence, considering all primitive variables, the state vector $\boldsymbol{W}$ is at least of dimension $n = 50\cdot 10^6$. Note that m snapshots $\boldsymbol{W} \in \mathbb{R}^{50\cdot 10^6}$ are required to construct an Isomap or POD based ROM.

To handle this amount of data, the grid is partitioned in p domains, where p is the number of available cores. Therefore, each core stores only the components of $\boldsymbol{W}$



(a) Offline costs depending on the snapshot size using 30 snapshots.

(b) Offline costs depending on the number of snapshots with a fixed snapshot size of 10^6.

Figure 6.2. "Average CPU time for the Isomap+I offline stage corresponding to setting up the ROM compared to the average POD+I offline stage CPU time (based on 10 runs each). Here, Isomap+I uses the 10 nearest neighbors to identify the data manifold and calculates a two dimensional embedding."(Franz et al. [36])

belonging to the corresponding domain:

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{W}_{dom_1} \\ \hline \boldsymbol{W}_{dom_2} \\ \hline \vdots \\ \hline \boldsymbol{W}_{dom_p} \end{pmatrix},$$

where dom_i denotes the ith domain.

The crux for running Isomap in parallel is to approximate the geodesic distances, or, more precisely, to calculate the Euclidean distances between neighboring snapshots based on the domain decomposition. Employing a brute force ansatz to determine the N nearest neighbors instead of the kd-tree, the squared Euclidean distances between each pair of snapshots on each core are calculated first:

$$d_{dom_k}(\boldsymbol{W}^i, \boldsymbol{W}^j) = \|\boldsymbol{W}^i_{dom_k} - \boldsymbol{W}^j_{dom_k}\|_2^2$$

for $k = 1, \ldots, p$ and $i, j = 1, \ldots, m$. To obtain the actual Euclidean distance between the pair of snapshots $(\boldsymbol{W}^i, \boldsymbol{W}^j)$, the sum of all distances $d_{dom_k}(\boldsymbol{W}^i, \boldsymbol{W}^j)$ is computed and squared afterwards:

$$\|\boldsymbol{W}^i - \boldsymbol{W}^j\|_2 = \sqrt{\sum_{k=1}^{p} d_{dom_k}(\boldsymbol{W}^i, \boldsymbol{W}^j)}.$$

Exploiting the pairwise distances $\|\boldsymbol{W}^i - \boldsymbol{W}^j\|_2$, the N-neighborhood graph is constructed and stored in a matrix of dimension $m \times m$ on each core. Since $m \ll n$, the remaining operations such as the shortest path computation or the application of MDS are conducted on each core simultaneously.

CHAPTER 7

Applications

The developed ROMs Isomap+I and Isomap+LSQ along with the adaptive sampling strategy, stated in Sections 4.5, 4.6 and 5.1, respectively, are now applied to 2D and 3D aerodynamic test cases. The first test case considers the steady transonic flow past the 2D NACA 64A010 airfoil, which is studied under varying angle of attack α and Mach number Ma. The second one deals with the steady transonic flow past the 3D XRF-1 fuselage-wing configuration, which is investigated under various wing deformations parameterized by five twist parameters. In common loads studies the C_p-distribution of the surface of an airfoil or wing is of interest. Hence, the evaluations are focused on this dimensionless quantity here. Unless otherwise stated, the following methods and functions are employed during the test cases:

The embedding is calculated via minimizing the loss function (4.4) with $w_{ij} = d_{ij}^2$. The k-nearest neighbors method is employed to determine the nearest neighbors for building the neighborhood graph and for the back-mapping. The optimal number of nearest neighbors to construct a neighborhood graph and hence to obtain approximations to the geodesic distances is then calculated via the proper stress function (4.13). Furthermore, variable-by-variable normalization is set to default, such that the data is scaled to $[0, 1]^{n_g}$ as described in the remark in Section 4.3.1.

For predictions, Isomap+I is coupled with the RBF interpolation using a thin plate spline (TPS) kernel (see Appendix A) augmented by a polynomial $\varphi \in \Pi_1$, $\varphi : \mathbb{R}^d \to \mathbb{R}$, where Π_1 is the space of polynomials of degree of at most one. “Prior to each interpolation process, the sample locations in the [parameter] space are scaled to the unit [hypercube], with the result that the input scaling is normalized and does not thwart the Isomap metric. The TPS kernel has been chosen for its good approximation quality and robustness based on best practice observations made by Zimmermann and Görtz [94].”(Franz et al. [36]) On the basis of the tests conducted in [35], the author also recommends the use of TPS as one of the best of the global basis function methods they considered. “Furthermore, [it is also preferred] for practical applications because other kernels such as Kriging with a Gaussian kernel require the tuning of hyperparameters, which may become expensive for many samples.”(Franz et al. [36])

The error functions E_{dist} and E_{rec} of the adaptive sampling strategies introduced in Section 5.1 are maximized via the popular derivative-free Nelder-Mead method [62, 91].

Notes on error measurement. ROM-based predictions often feature offsets to the reference solution up- or downstream of the shock region. Since there typically is only a small number of grid points in the shock region compared to the remaining part of the surface, when measuring the error with respect to the 1-norm or 2-norm these small offsets accumulate to a larger error, which blurs the actual accuracy of the shock predictions. The objective of this thesis is, however, to construct a ROM featuring an improved shock prediction behavior (see Section 1.2). Hence, for a prediction at the parameter combination $\boldsymbol{p} \in \mathcal{P}$ the error is only computed at grid points where the difference of the predicted C_p-distribution to the reference solutions is larger than a threshold $\theta_{\boldsymbol{p}}$. Let $\boldsymbol{W}^{\text{ref}}(\boldsymbol{p}) \in \mathbb{R}^n$ and $\boldsymbol{W}^*(\boldsymbol{p}) \in \mathbb{R}^n$ denote the reference solution and the ROM-based prediction at $\boldsymbol{p} \in \mathcal{P}$, respectively, where n is the product of the number of grid points and the number of variables, i. e. $n = n_g n_v = n_g$. Hence, the i-th component of a solution $\boldsymbol{W}^{\text{ref}}(\boldsymbol{p})$ or prediction $\boldsymbol{W}^*(\boldsymbol{p})$ corresponds to the grid point i. Define $\mathcal{I}_R := \{i \in \{1, \ldots, n_g\} \mid |\boldsymbol{W}_i^{\text{ref}} - \boldsymbol{W}_i^*| > \theta_{\boldsymbol{p}}\}$, where the subscripts R and i specify the ROM yielding the prediction $\boldsymbol{W}^*(\boldsymbol{p})$ and the i-th component of the vector $\boldsymbol{W}^*(\boldsymbol{p})$, respectively. Then the error at a prediction point $\boldsymbol{p} \in \mathcal{P}$ is computed as follows:

$$\operatorname{err}(\boldsymbol{W}^{\text{ref}}(\boldsymbol{p}), \boldsymbol{W}^*(\boldsymbol{p})) = \frac{\sum_{i \in \mathcal{I}_R} |\boldsymbol{W}_i^{\text{ref}} - \boldsymbol{W}_i^*|}{\sum_{i \in \mathcal{I}_R} |\boldsymbol{W}_i^{\text{ref}}|} \tag{7.1}$$

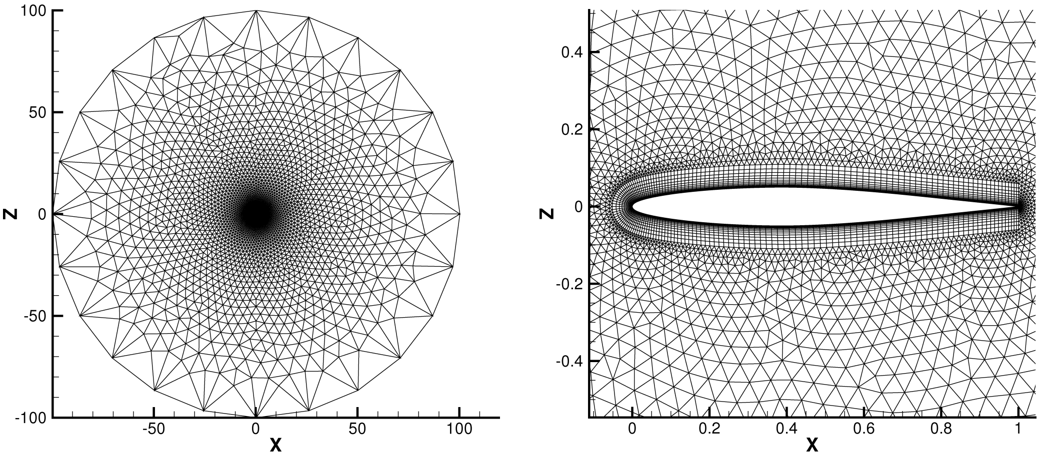
According to engineering experience and a good trade-off between the actual error and an error focusing on the shock prediction, the threshold $\theta_{\boldsymbol{p}}$ is set to 1% of the total range of the C_p-distribution of the respective reference solution $\boldsymbol{W}^{\text{ref}}$, i. e.

$$\theta_{\boldsymbol{p}} = (\max_i \boldsymbol{W}_i^{\text{ref}} - \min_i \boldsymbol{W}_i^{\text{ref}}) \cdot 0.01. \tag{7.2}$$

For a fair comparison, the union of the different index sets $\mathcal{I}_{R_1}, \mathcal{I}_{R_2}, \ldots, \mathcal{I}_{R_l}$ occurring for various ROMs $R_1, R_2, \ldots, R_l$ is employed to calculate the error (7.1).

7.1. NACA 64A010 airfoil

In this section, the flow past the two-dimensional NACA 64A010 airfoil in the transonic flow regime is considered. The geometry of the airfoil is shown in Figure 7.1(b).



(a) View of the entire flow field. **(b)** Detailed view close to the surface.

Figure 7.1. Computational grid for the NACA 64A010 airfoil.

The hybrid unstructured grid features 21,454 grid points, including 400 surface grid points, and is depicted in Figure 7.1.

To generate a data manifold W to capture the characteristics of the flow past the airfoil under varying angle of attack, α, and Mach number, Ma, a LHS [33] of $m = 30$ samples is performed in the parameter space $\mathcal{P} = [4°, 10°] \times [0.74, 0.82] \subset \mathbb{R}^2$, defining the set of parameters $P \in \mathcal{P}$ (see Figure 7.2). The corresponding viscous flow solution snapshots $\boldsymbol{W}(\boldsymbol{p})$, $\boldsymbol{p} \in P$, are computed with the RANS solver TAU (see Section 2.2) using the negative Spalart-Allmaras (SA) one-equation turbulence model [2], whereby the normalized density residual is reduced by seven orders of magnitude for each solution. The Reynolds number is fixed throughout at a value of $\mathrm{Re} = 7,500,000$.

All computations were conducted sequentially on the same standard desktop computer endowed with an Intel® Xeon® E3-1270 v3 Processor (8M Cache, 3.50 GHz) and 32 GB RAM. Computing a full CFD solution for this test case until the density residual converged by seven orders of magnitude took 474 iterations or 63 CPU seconds on average (based on the computed reference solutions in Section 7.1.1).

The set up of this test case is the same as in [36] published previously by Franz et al., but the results obtained by Isomap+I may differ slightly since functionality developed later like various loss and stress functions to calculate the embedding and to determine the neighborhood size for the neighborhood graph, respectively, are exploited now. In addition, the adaptive method for choosing the neighborhood size for predicting full-order solutions is also applied so that Isomap+I is a fully automated process after selecting a proper stress and loss function.

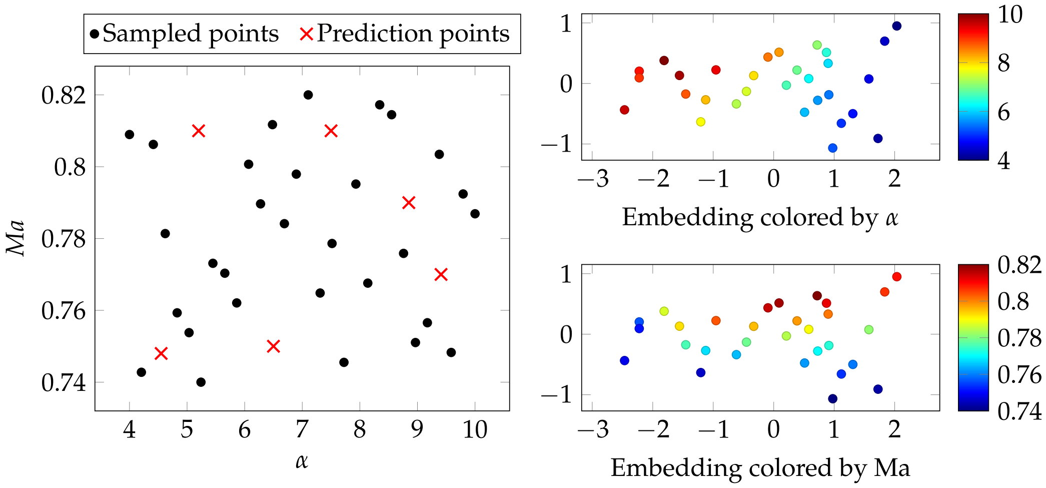


Figure 7.2. Left side: Locations of the snapshots and the prediction points in the α-Ma-space for the NACA 64A010 test case. **Right side:** Representatives within the embedding space colored corresponding to the angle of attack α (top) and the Mach number Ma (bottom).

7.1.1. Isomap with interpolation

Based on the DoE obtained by LHS, the prediction capability of Isomap+I is investigated on the basis of TAU reference solutions at arbitrary untried parameter combinations $p \in \mathcal{P} \setminus P$ (see Figure 7.2). "Since in this test case two varying parameters are considered, the full-order solution manifold $\mathcal{W} = \{\boldsymbol{W}(\alpha, \mathrm{Ma}), \quad (\alpha, \mathrm{Ma}) \in \mathcal{P}\} \subset \mathbb{R}^{400}$ is of [intrinsic] dimension two. Making use of this *a priori* information, the Isomap algorithm is applied to the surface C_p-distribution vectors of the 30 full-order snapshots $[\boldsymbol{W}(\boldsymbol{p}^i) \in \mathbb{R}^{400}]$ to compute a $(d = 2)$-dimensional embedding consisting of 30 representatives $\boldsymbol{y}^i \in \mathbb{R}^2$."(Franz et al. [36]) To ensure an optimal embedding regarding the stress function, the number of nearest neighbors for constructing the neighborhood graph is determined to be five (see Figure 7.3). The adaptive method for choosing the number of nearest neighbors for the back-mapping is utilized with a minimum number of three neighbors and a maximum number of 10 neighbors. The lower and upper bounds for the weights a_i of the linear combination yielding a CFD prediction are set to $\xi_l = -0.2$ and $\xi_u = 1.2$, respectively. Summarizing, Isomap+I is performed with the following parameters:

m	n	d	N	N_{rec}
30	400	2	5	$3-10$

The variables m, n, d, N and N_{rec} denote the number of snapshots, the dimension of the full-order solution space, the dimension of the embedding space, the number of

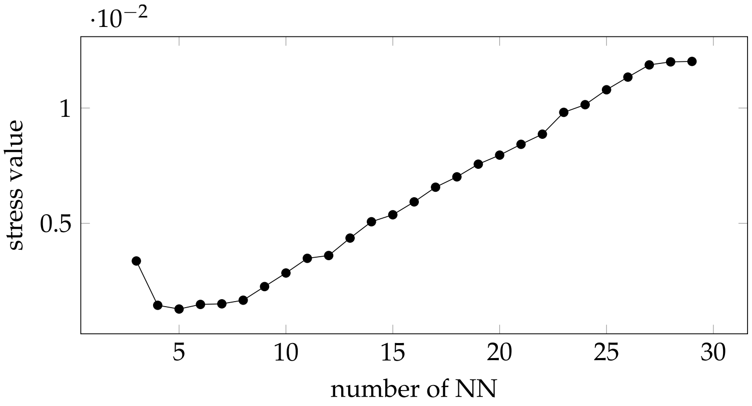


Figure 7.3. Stress function applied to the NACA 64A010 test case restricted to surface C_p data.

nearest neighbors used to build the neighborhood graph and the number of nearest neighbors employed by the back-mapping, respectively.

Since Isomap is originally a tool for data analysis, the embedding can be exploited to gain more details about the test case. Top right in Figure 7.2, the embedding is colored corresponding to the value of the angle of attack α. As it can be seen, the larger the difference of α-values the bigger is the distance in the embedding space, i. e., the surface C_p solutions are more different. However, compared to the plot downright in Figure 7.2, where the embedding is colored corresponding to the value of the Mach number Ma, larger changes of the Mach number Ma have less influence on the solution than changes of α. Hence, regarding the parameter space $\mathcal{P}$, the flow solutions are more sensitive under varying angle of attack at a fixed Mach number than vice versa.

For comparison, a global POD of the 30 full-order surface C_p snapshots is performed, yielding a basis consisting of 29 orthonormal POD modes of dimension 400, since the mean of the snapshots is subtracted (see Section 3.1). Additionally, a reduced global POD, hereafter referred to as redPOD+I, consisting of five modes is also conducted, which corresponds roughly to the mean value of the number of nearest neighbors employed by an Isomap based prediction in this case. For computing flow predictions, each POD model is combined with a TPS interpolation scheme as introduced in Section 3.3.

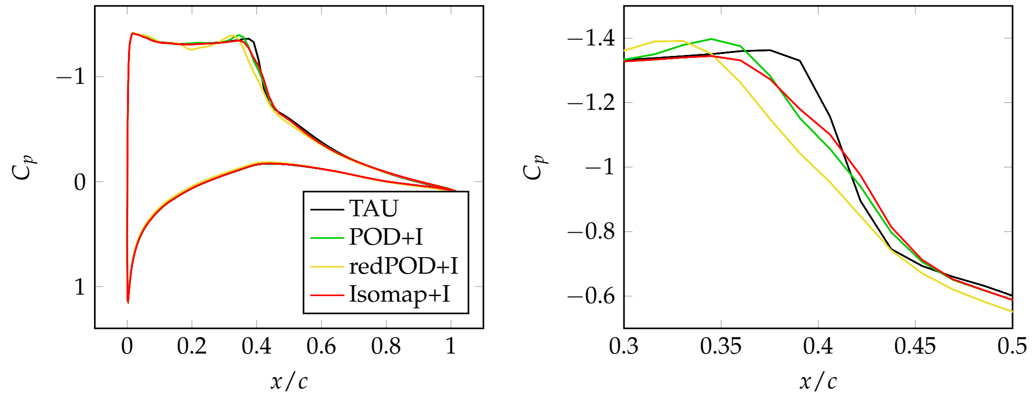
The Isomap+I and both POD based models were built in less than 1.54 and 0.01 CPU seconds, respectively, including the data processing, setting up the TPS model and, in the case of Isomap, the computation of the proper number of nearest neighbors and, due to the loss function, additional execution of variable alternation. It should be mentioned that the CPU time of building an Isomap+I ROM is reduced to

α	Ma	Isomap+I	POD+I	redPOD+I
4.55	0.748	$3.0803 \cdot 10^{-2}$	$3.3143 \cdot 10^{-2}$	$6.5257 \cdot 10^{-2}$
7.5	0.81	$1.6189 \cdot 10^{-2}$	$5.4215 \cdot 10^{-2}$	$4.8380 \cdot 10^{-2}$
8.85	0.79	$2.1819 \cdot 10^{-2}$	$6.7081 \cdot 10^{-2}$	$4.9849 \cdot 10^{-2}$
9.41	0.77	$3.3689 \cdot 10^{-2}$	$5.9239 \cdot 10^{-2}$	$7.2976 \cdot 10^{-2}$
5.2	0.81	$5.4371 \cdot 10^{-2}$	$5.2378 \cdot 10^{-2}$	$7.0818 \cdot 10^{-2}$
6.5	0.75	$7.0572 \cdot 10^{-2}$	$7.8926 \cdot 10^{-2}$	$7.3781 \cdot 10^{-2}$

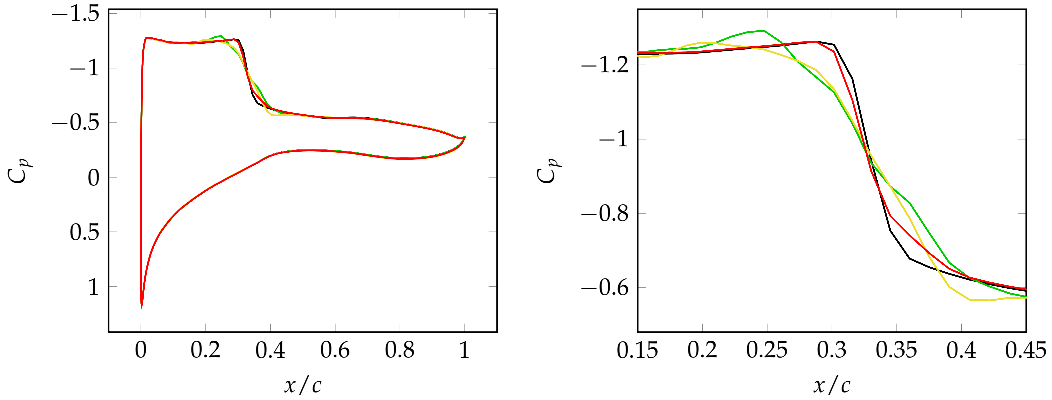
Table 7.1. Errors in terms of equation (7.1) between the TAU reference surface C_p solutions and the surface C_p predictions obtained by Isomap+I, POD+I and redPOD+I at various parameter combinations.

0.03 CPU seconds if the classical MDS solution is requested and to less than 0.01 CPU seconds if additionally the number of nearest neighbors is chosen by hand (see Section 6.2). However, the online prediction of a surface solution at an untried parameter combination $\boldsymbol{p} \in \mathcal{P} \setminus P$ was instantaneous for all ROMs, whereas a full CFD solution took 63 CPU seconds on average to compute with TAU.

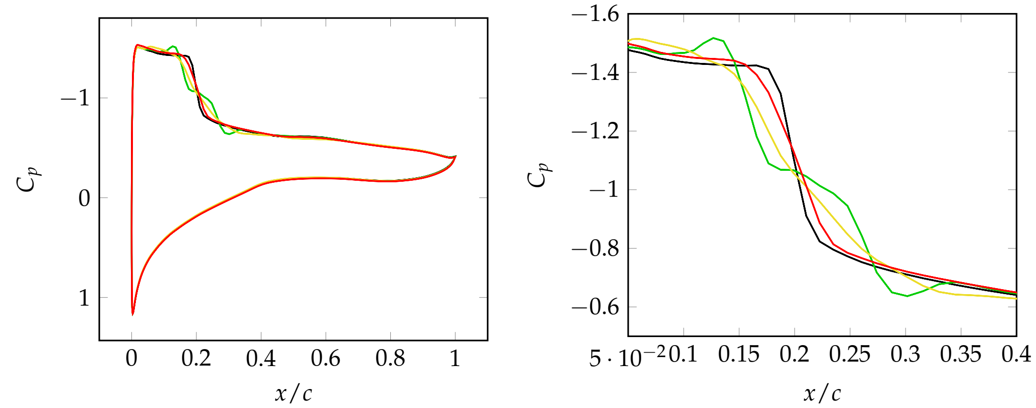
The resulting surface C_p-distributions predicted by Isomap+I, POD+I and redPOD+I for various parameter combinations $\boldsymbol{p} \in \mathcal{P} \setminus P$ are compared to the corresponding TAU reference solutions in Figures 7.4-7.5, where the C_p-distributions are plotted against the normalized chord length x/c. As it can be seen in Figures 7.4(a)-7.5(c) and Figure 7.5(a), Isomap+I quite accurately predicts the shape and the location of the shock calculated by TAU, whereas POD+I and redPOD+I fail to predict accurate solutions. Both POD-based methods have bumps up- or downstream of the shock and, at least in the case of POD+I, most predictions show the typical step-function behavior often observed for POD-based shock predictions. This offset is reflected in the errors in terms of equation (7.1) between the TAU reference solutions and the corresponding flow predictions obtained by Isomap+I, POD+I and redPOD+I stated in Table 7.1. At all prediction locations except one it holds that the Isomap+I prediction error is smaller than the prediction errors of both POD based ROMs. In particular, the Isomap+I based error is up to three times smaller than the POD based errors. However, in Figure 7.5(b) and Figure 7.5(c) it can be seen that all ROMs fail to predict an accurate shock at $\boldsymbol{p} = (7.5°, 0.81)$ and $\boldsymbol{p} = (8.85°, 0.79)$. This indicates that the information content of the data manifold $W = \{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^m\}$ is insufficient at this parameter combinations for all investigated ROMs. A remedy may be given by a CFD-enhanced prediction or by an adaptively sampled DoE for the specified parameter space $\mathcal{P}$. These approaches are investigated in the Sections 7.1.2 and 7.1.3.



(a) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (4.55°, 0.748)$.

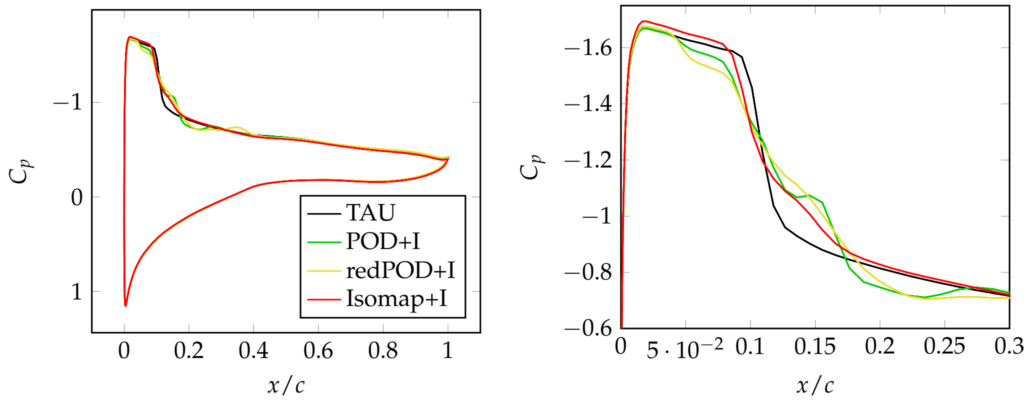


(b) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (7.5°, 0.81)$.

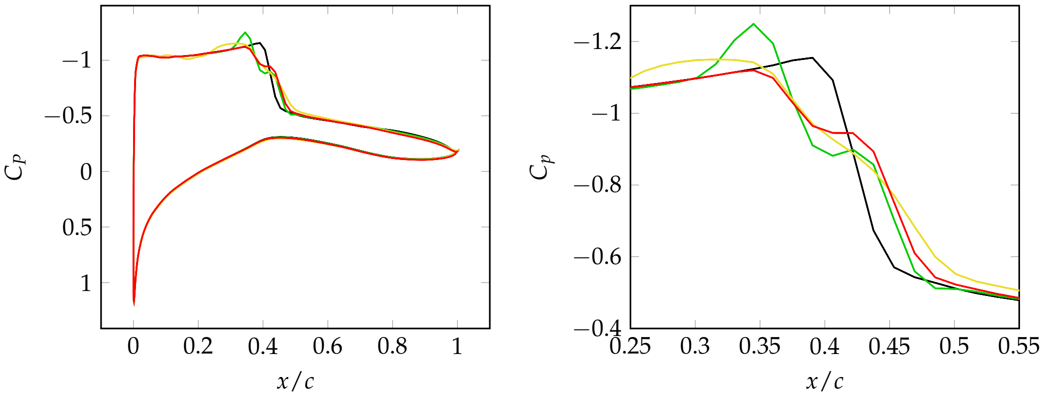


(c) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (8.85°, 0.79)$.

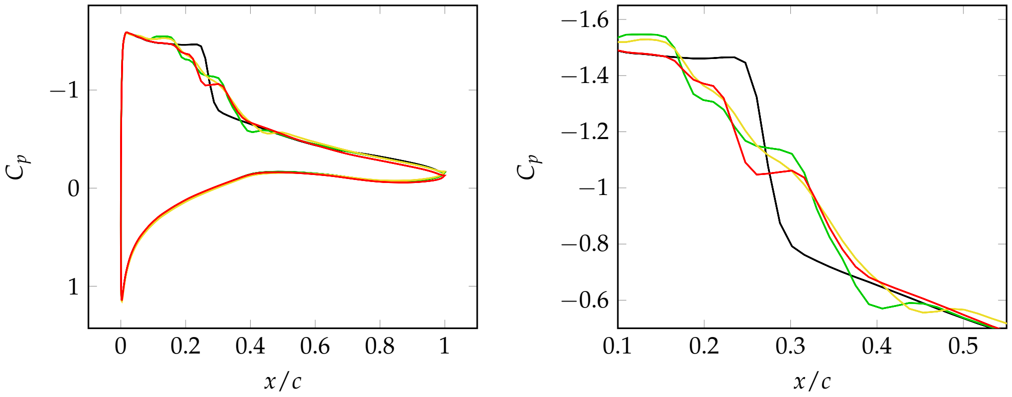
Figure 7.4. Surface C_p predictions at various (α, Ma)-combinations, where the upper and lower curves correspond to the suction and pressure side of the airfoil, respectively. On the left side, the entire surface C_p-distribution is depicted. On the right side, a detailed view of the shock of the particular distribution is given.



(a) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (9.41°, 0.77)$.



(b) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (5.2°, 0.81)$.



(c) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (6.5°, 0.75)$.

Figure 7.5. Surface C_p predictions at various (α, Ma)-combinations, where the upper and lower curves correspond to the suction and pressure side of the airfoil, respectively.

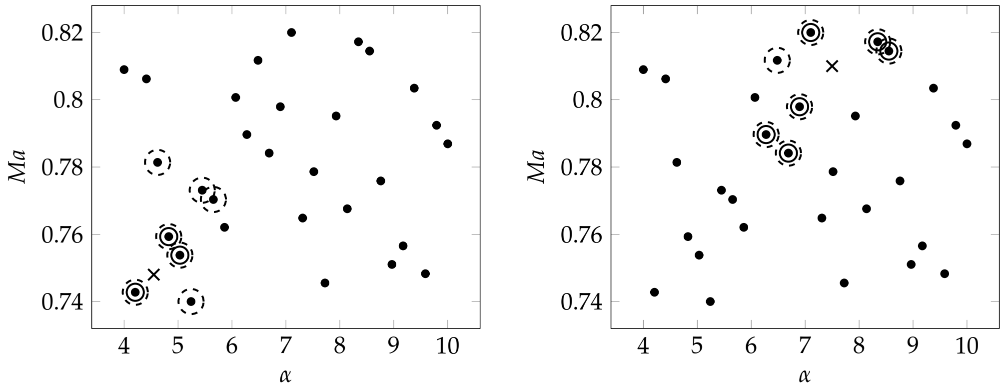
Note that the Isomap+I predictions only rely on local characteristics given by the nearby flow solutions. The corresponding (α, Ma)-combinations for all predictions are encircled in Figure 7.6. "The solutions at these parameter combinations are assumed to be the nearest neighbors to the requested solution on the data manifold. Note that nearest neighbors on the manifold are not necessarily nearest neighbors in the α-Ma space [or normalized α-Ma space]."(Franz et al. [36])

Remark. For comparison, alternative methods like local POD bases based on the snapshots belonging to the six nearest neighbors determined by Isomap and based on the six nearest Euclidean neighbors in the normalized (α, Ma)-parameter space were also considered. But "[e]ven when restricting the snapshots basis to the same local features as determined by Isomap+I, the predictions by POD+I do not achieve the same quality in terms of shock resolution as the predictions obtained by Isomap+I."(Franz et al. [36]) The results can be found in [36].

7.1.2. Isomap with residual optimization

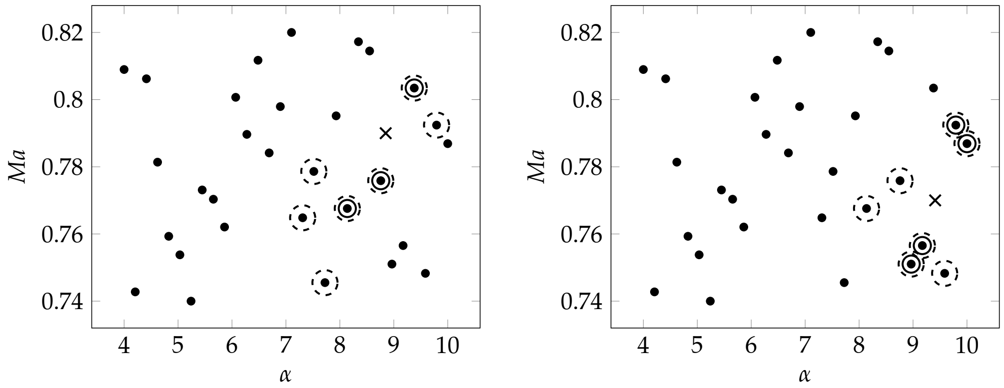
Employing interpolation-based non-intrusive ROMs like Isomap+I and POD+I, the predictions obtained by these methods may be inaccurate or even unphysical at some parameter combinations (see Figures 7.5(b) and 7.5(c)). In order to improve these predictions, the coefficient vectors of the interpolated predictions are now exploited as starting vectors for the residual-based LSQ-ROMs derived in Sections 4.6 and 3.4. As introduced in Section 4.6, each ROM exploits the farfield boundary values of the corresponding interpolation-based prediction to define an upper bound for the farfield boundary constraint of the constrained optimization (4.28). Furthermore, as suggested in [92, Observation 1] for unconstrained POD+LSQ, only the total energy residual values belonging to the 20% smallest cells are taken into account. Since the computational grid features the smallest cells exactly close to the surface, the defect of the governing equations is minimized in the region of interest. Sequential least squares programming (SLSQP) [50] is applied to optimize the coefficients of the nearest snapshots for the Isomap+LSQ method (cf. the constrained minimization problem (4.28)) and to optimize the coefficients of the POD modes, respectively.

CFD flow-field solutions including the values of all primitive variables, namely the density, the velocities, the pressure and the turbulence, are required to evaluate the TAU residual. The C_p-distribution is also included in the snapshots so that the ROMs are capable of predicting C_p values directly. Otherwise, for each prediction the C_p-distribution has to be computed in the online phase, which would lead to unnecessary extra CPU times. Hence, Isomap+LSQ, POD+LSQ and redPOD+LSQ are based on 30 snapshots of dimension $n_g \cdot n_v = 171632$, increasing the building time of both



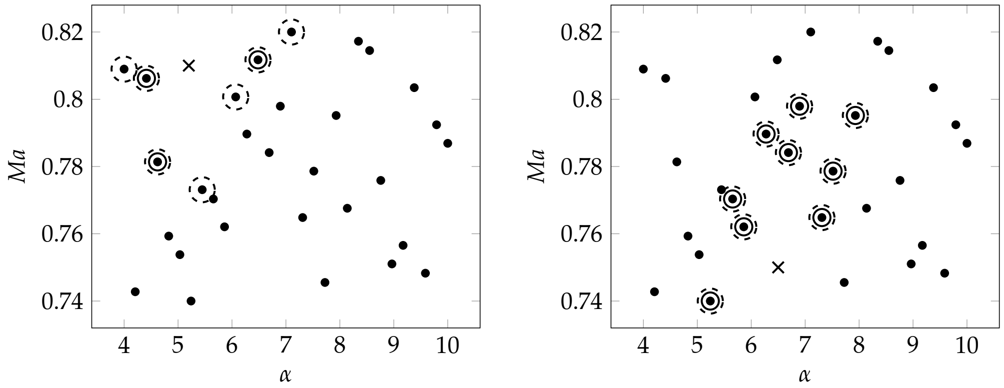
(a) $(\alpha, \mathrm{Ma}) = (4.55°, 0.748)$.

(b) $(\alpha, \mathrm{Ma}) = (7.5°, 0.81)$.



(c) $(\alpha, \mathrm{Ma}) = (8.85°, 0.79)$.

(d) $(\alpha, \mathrm{Ma}) = (9.41°, 0.77)$.



(e) $(\alpha, \mathrm{Ma}) = (5.2°, 0.81)$.

(f) $(\alpha, \mathrm{Ma}) = (6.5°, 0.75)$.

Figure 7.6. Parameter combinations of the snapshots employed for the flow prediction at different parameter combinations as determined by Isomap+I for the NACA 64A010 test case. The crosses, the points, the encircled points and the dashed encircled points denote the prediction points, the sample points, the employed nearest neighbors on the manifold by Isomap+I and the employed nearest neighbors on the manifold by Isomap+LSQ, respectively.

POD based ROMs to 0.59 CPU seconds. However, for calculating the embedding, Isomap+LSQ takes only the surface C_p-distribution into account, as before. Therefore, the costs of building Isomap+LSQ are the same as for Isomap+I. Employing only three nearest neighbors for the back-mapping as before, the Isomap+LSQ method fails to improve the predictions. Hence, the minimum number for nearest neighbors of the back-mapping is increased from three to seven, so that the optimization problem (4.28) gains more degrees of freedom. The maximum number for nearest neighbors of the back-mapping, however, is not changed. Thus, the initial solutions for the optimization obtained by Isomap+I are calculated by using the 7-10 nearest neighbors, whose corresponding parameters are denoted by the dashed encircled points in Figure 7.6. Note that this initial solution may differ from the Isomap+I predictions obtained in Section 7.1.1 since it is now based on at least seven neighbors. After evaluating the TAU residual, the discrepancy in the total energy, restricted to the 20% smallest cells, is exploited by the objective function to optimize the coefficient vector $\boldsymbol{a} \in \mathbb{R}^d$, until the tolerance for termination ($tol = 10^{-8}$) is reached. Here, d has the values $d \in \{7, 8, 9, 10\}$, $d = 29$ or $d = 7$ for Isomap+LSQ, POD+LSQ or redPOD+LSQ, respectively. Note that for reasons of clarity the results obtained by redPOD+LSQ are dropped, since the optimization of the seven POD coefficients just slightly improved the initial prediction based on seven modes. The predictions are still far off the corresponding reference solutions compared to the predictions of the remaining ROMs and, in particular, compared to the predictions of a full POD based ROM (cf. [92, Section 4]). In conclusion, a POD-based residual optimization using only a few POD modes does not achieve the accuracy of an Isomap-based residual optimization employing a set of proper snapshots of equal number. This may be due to the fact that a reduced POD based ROM truncates the solution space, whereas Isomap based ROMs choose a set of proper snapshots and hence a proper solution space of low-dimension.

The CFD enhanced predictions at selected parameter combinations $\boldsymbol{p} \in \mathcal{P} \setminus P$ are shown in Figure 7.7. As it can be seen, the gradient of shocks are corrected (see Figures 7.7(a)-7.7(b)) and unphysical predictions like solutions comprising step-like shocks are adjusted (see Figure 7.7(b)). Due to the larger number of DoF, POD+LSQ yields a more accurate solution at the prediction point $(\alpha, \mathrm{Ma}) = (5.2°, 0.81)$, but, however, the larger DoF is also reflected in CPU times. The errors in terms of equation (7.1) between the TAU reference solutions and the corresponding flow predictions obtained by Isomap+LSQ, POD+LSQ and redPOD+LSQ are stated in Table 7.2. Comparing the error values to the previous ones in Table 7.1, an improvement of all but two Isomap based prediction is observed, whereas this is not the case for the POD+LSQ based predictions and not for the redPOD+LSQ based predictions as well.

α	Ma	Isomap+LSQ	POD+LSQ	redPOD+LSQE
4.55	0.748	$2.2353{\cdot}10^{-2}$	$5.4220{\cdot}10^{-2}$	$5.6458{\cdot}10^{-2}$
7.5	0.81	$2.2207{\cdot}10^{-3}$	$1.1400{\cdot}10^{-2}$	$5.8223{\cdot}10^{-2}$
8.85	0.79	$1.8242{\cdot}10^{-2}$	$6.6657{\cdot}10^{-2}$	$4.1856{\cdot}10^{-2}$
9.41	0.77	$2.5258{\cdot}10^{-2}$	$3.9440{\cdot}10^{-2}$	$4.8736{\cdot}10^{-2}$
5.2	0.81	$4.6575{\cdot}10^{-2}$	$4.0034{\cdot}10^{-2}$	$7.5198{\cdot}10^{-2}$
6.5	0.75	$6.7803{\cdot}10^{-2}$	$7.5089{\cdot}10^{-2}$	$8.2010{\cdot}10^{-2}$

Table 7.2. Errors in terms of equation (7.1) between the TAU reference surface C_p solutions and the extracted surface C_p predictions obtained by Isomap+LSQ, POD+LSQ and redPOD+LSQ at various parameter combinations.

Due to the smaller number of DoF of the Isomap based optimization problem, which is 7-10 for Isomap+LSQ and 29 for POD+LSQ, the optimization performs at least 7× faster compared to the POD+LSQ ROM and 2-4.5× faster than computing a full CFD solution (average CPU time of 63s). The large number of DoF of POD+I appears to slow down the constrained optimization via SLSQP. Remedy may be given by replacing the SLSQP with a specific constrained LSQ-solver as in [95]. A detailed listing of the CPU times, number of iterations, number of function evaluations and the speed-up factor is given in Table 7.3.

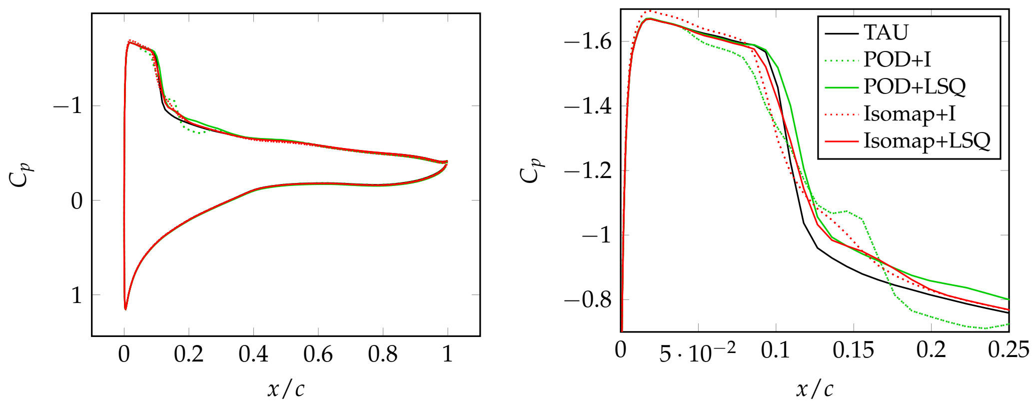
However, at $(\alpha, \mathrm{Ma}) = (6.5^\circ, 0.75)$ all ROMs still feature an inaccurate prediction of the shock, but at least it is smoothened. Consequently, the sampled data manifold W does not contain sufficient informations of the true manifold $\mathcal{W}$ area-wide in order to obtain accurate predictions by the investigated ROMs for all $\boldsymbol{p} \in \mathcal{P}$. A remedy may be given by an adaptively sampled DoE for the specified parameter space $\mathcal{P}$ (see Section 7.1.3).

7.1.3. Adaptive sampling

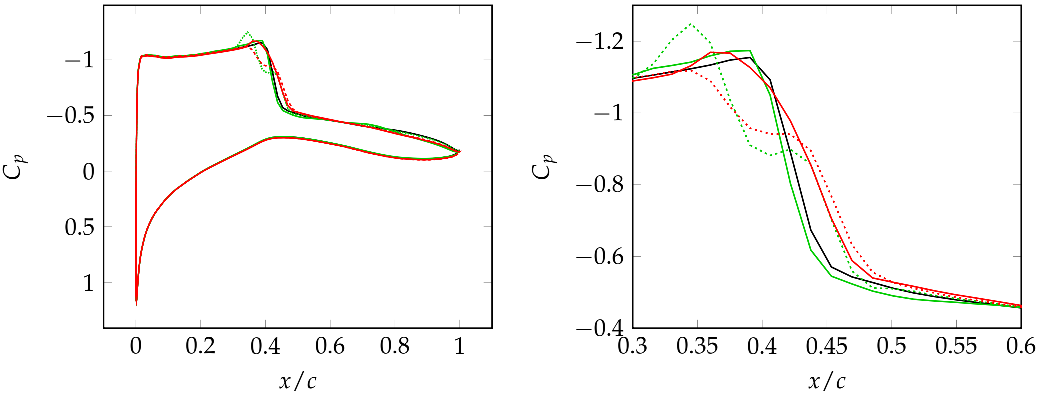
Concluding from the results above, either the number of snapshots has to be increased or a different DoE should be employed to obtain ROMs capable of yielding accurate predictions for arbitrary $\boldsymbol{p} \in \mathcal{P}$. Since increasing the number of snapshots is expensive, an adaptively sampled DoE as introduced in Chapter 5 is preferred.

To demonstrate the ability of MDE and HYE to expand the description of the manifold even if the given input data set only describes a part of the manifold, the initial DoE is generated by a Halton sequence [43] of five points, whereby neither of them lies on the boundary of $\mathcal{P}$.

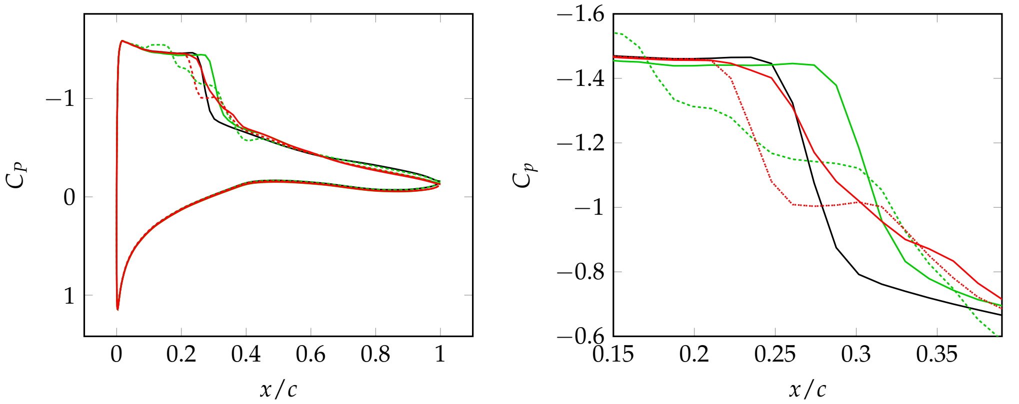
Remark. Since the MDE based adaptive sampling only depends on the description of



(a) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (9.41°, 0.77)$.



(b) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (5.2°, 0.81)$.



(c) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (6.5°, 0.75)$.

Figure 7.7. Surface C_p predictions at various (α, Ma) combinations, where the upper and lower curves correspond to the suction and pressure side of the airfoil, respectively. On the left side, the entire surface C_p-distribution is depicted. On the right side, a detailed view of the shock of the particular distribution is given.

Isomap+LSQ CPU times (s)	 iter	 f_eval	POD+LSQ CPU times (s)	 iter	 f_eval	speed-up factor
19.57	16	166	187.32	54	1,726	9.57
13.69	11	118	149.02	43	1,370	10.89
15.04	12	127	161.67	46	1,464	10.75
16.36	13	135	192.46	54	1,717	11.76
19.63	16	160	192.08	53	1,684	9.79
30.82	20	235	219.93	60	1,903	7.14

Table 7.3. CPU times, number of iterations and function evaluations of conducting Isomap+LSQ and POD+LSQ. The speed-up gained by performing Isomap+LSQ instead of POD+LSQ is stated in the last column. The rows are arranged as in Table 7.2.

the manifold instead of an error yielded by a specific ROM, it should also improve the prediction of POD-based methods since these ROMs also rely on a proper description of the manifold.

The setup of the sampling process is as follows. As before, the surface C_p-distribution is of interest, so that the input data for Isomap is restricted to this quantity. The loss and stress functions are chosen as before, however, in each iteration, the stress function has to be exploited to determine the number of nearest neighbors for the embedding. The adjustment of the number of nearest neighbors in each iteration is limited to at most 2, i. e., $|N - N_{new}| \leq 2$, to ensure a “smooth” change of the detected manifold. The chosen numbers of nearest neighbors are shown in Figure 7.8. Both sampling strategies employ only the nearest neighbor, i. e. $|\mathcal{I}| = 1$, to evaluate the objective function (5.2), whereby HYE exploits E_{rec} in every third iteration. All remaining details were stated in the introduction to this chapter.

Starting with an initial sampling of five parameter combinations $\boldsymbol{p}^i$, $i = 1, \ldots, 5$, depicted by the red points in Figure 7.9 and calculated by a Halton sequence, and snapshots $\boldsymbol{W}^i = \boldsymbol{W}(\boldsymbol{p}^i) \in \mathbb{R}^{400}$, 25 iterations are performed to obtain a final sampling of 30 sample points along with the corresponding snapshots. The mean relative errors, the standard deviations and the maximum relative errors of the sampling processes of the MDE, HYE and, for comparising purposes, a Halton sequence based sampling are shown in Figure 7.10. 2500 TAU reference solutions, calculated at uniformely distributed locations in the parameter space, are used to compute the relative errors of the predictions by Isomap+I and POD+I at these locations. All errors are measured regarding the surface C_p values, which is the variable of interest. As can be seen in Figure 7.10, all three sampling methods result in nearly the same mean relative error after 25 iterations for both ROMs, but employing MDE and HYE lead

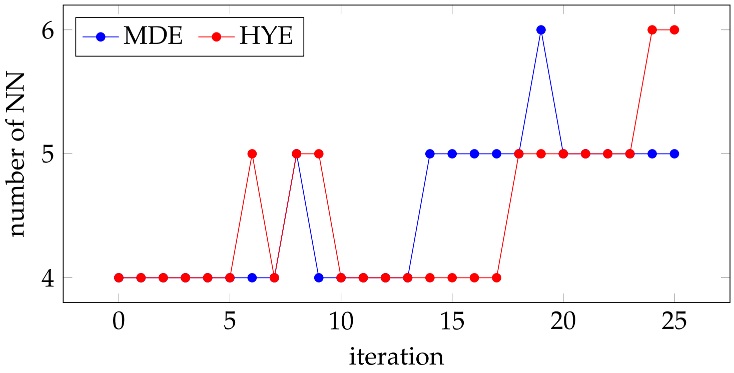


Figure 7.8. Employed number of nearest neighbors to construct the neighborhood graph in each iteration. The values are determined by the stress function, which was selected for the NACA 64A010 test case in advance.

to an Isomap+I and POD+I ROM with less standard deviation. Hence, the maximum relative error also shown in Figure 7.10 is smaller for both adaptive methods than for the Halton sampling. The final samplings of the MDE, the HYE and the Halton sequences based sampling are shown in Figure 7.9. For comparising purposes, the LHS of 30 sample points employed in Sections 7.1.1 and 7.1.2 and two *full factorial* designs, which are denoted by “FF”, followed by the number of grid points in each dimension, are also depicted. The relative error of an Isomap+I prediction at a specific parameter combination is shown by the contour in each plot. As already seen in Figure 7.10, the sampling strategies developed in Section 5 yield samplings with a smaller change of the relative errors than in both random samplings. Hence the maximum relative error is closer to the mean relative error, which leads to a more reliable global ROM with less outliers in prediction accuracy. This assertion can also be taken from Figure 7.10. The exact values of the mean relative errors, standard deviations and maximum relative errors after 25 iterations are stated in Table 7.4. Comparing “FF 6x5” and ”FF 5x6”, it can be seen, that unlike the adaptive methods, some knowledge of the behavior of the solution regarding the parameter space may be necessary to set up a full factorial sampling, otherwise it can lead to a larger mean relative error.

The embeddings of the final samplings are shown in Figure 7.11. As aspired by MDE, the embedding of the corresponding sampling is quite evenly distributed. This also holds for the embedding of the sampling obtained by HYE, even if E_{rec} is applied in every third iteration. In contrast, the embeddings of both random samplings feature close-by points, which may lead to redundant information content. The full factorial designs are more evenly distributed than the random samplings, but the density still varies within the embedding space.

method	mean rel. error	std. deviation	max. rel. error
MDE	$2.3347{\cdot}10^{-2}$	$1.6616{\cdot}10^{-2}$	$9.2956{\cdot}10^{-2}$
HYE	$2.1903{\cdot}10^{-2}$	$1.0320{\cdot}10^{-2}$	$5.3337{\cdot}10^{-2}$
Halton	$2.6670{\cdot}10^{-2}$	$2.7398{\cdot}10^{-2}$	$2.3016{\cdot}10^{-1}$
LHS	$3.1262{\cdot}10^{-2}$	$2.6257{\cdot}10^{-2}$	$1.8009{\cdot}10^{-1}$
FF6x5	$2.3990{\cdot}10^{-2}$	$1.5855{\cdot}10^{-2}$	$1.0257{\cdot}10^{-1}$
FF5x6	$2.6235{\cdot}10^{-2}$	$1.7485{\cdot}10^{-2}$	$1.0324{\cdot}10^{-1}$

Table 7.4. The mean relative error, its standard deviation and the maximum relative error after a full sampling process of various sampling strategies/designs for the NACA 64A010 test case.

The error plots of E_{dist} and E_{rec} after 6, 14 and 19 iterations of HYE are compared to the actual relative error in Figure 7.12. Since MDE and HYE employ error heuristics, i. e. E_{dist} and E_{rec}, it is not expected that $\boldsymbol{p}^*$ matches the parameter combination with the largest model error in each iteration. However, in iteration 16 the error plot of E_{rec} is similar to the actual error and $\boldsymbol{p}^*$, denoted by the red diamond, is contained in the region of largest error. The error surfaces of E_{dist} in iteration 6 and 19 have different maxima compared to the actual error, but the optima of E_{dist} match regions where the actual error is still large.

The ROM predictions at $(\alpha, \mathrm{Ma}) = (6.5^\circ, 0.75)$ based on the HYE sampling (see Figure 7.13(b)) are shown in Figure 7.13(a). As it can be seen, both predictions are greatly improved (cf. Figures 7.5(c) and 7.7(c)). The Isomap+I prediction matches the reference solution with high accuracy and the POD+I based prediction only shows a small mismatch upstream of the shock. In conclusion, the adaptive sampling strategy selected proper sampling points in the parameter space where poor predictions have been observed before.

7.2. XRF-1 fuselage-wing configuration

Now, the flow past the three-dimensional XRF-1 fuselage-wing configuration in the transonic flow regime is considered. The geometry of the configuration and the underlying structured grid featuring 784,384 grid points, including 19,211 surface grid points, are depicted in Figure 7.14. For this test case, the Mach number and the Reynolds number are fixed at $\mathrm{Ma} = 0.83$ and $\mathrm{Re} \approx 43.4 \cdot 10^6$, respectively. Furthermore, a target lift coefficient of $C_l = 0.5$ is prescribed and five twist sections are defined as parameters. The twist sections and the maximum twist in different rotation directions are shown in Figure 7.15.

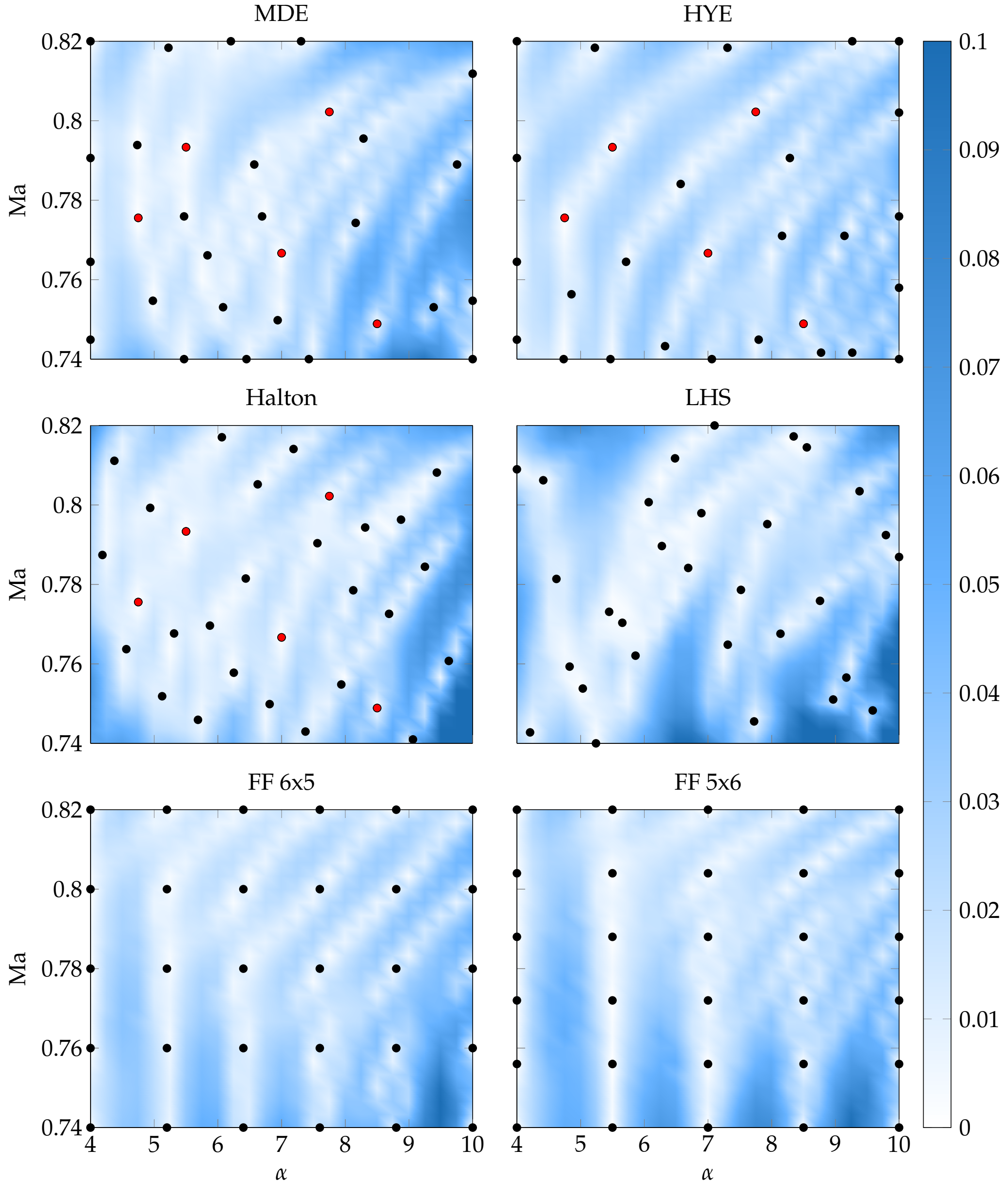


Figure 7.9. Final samplings obtained by various sampling methods and DoEs. In the upper Figures from left to right, the red points denote the initial sampling obtained by a Halton sequence and the black points denote the adaptively sampled parameter points by MDE and HYE, respectively. In the middle Figures from left to right, the continued Halton sequence up to 30 parameter combinations and the LHS yielding 30 different parameter combinations (cf. Sections 7.1.1 and 7.1.2) are shown, respectively. In the bottom row, two full factorial samplings are depicted. The contour in each plot depicts the relative error of the Isomap+I predictions.

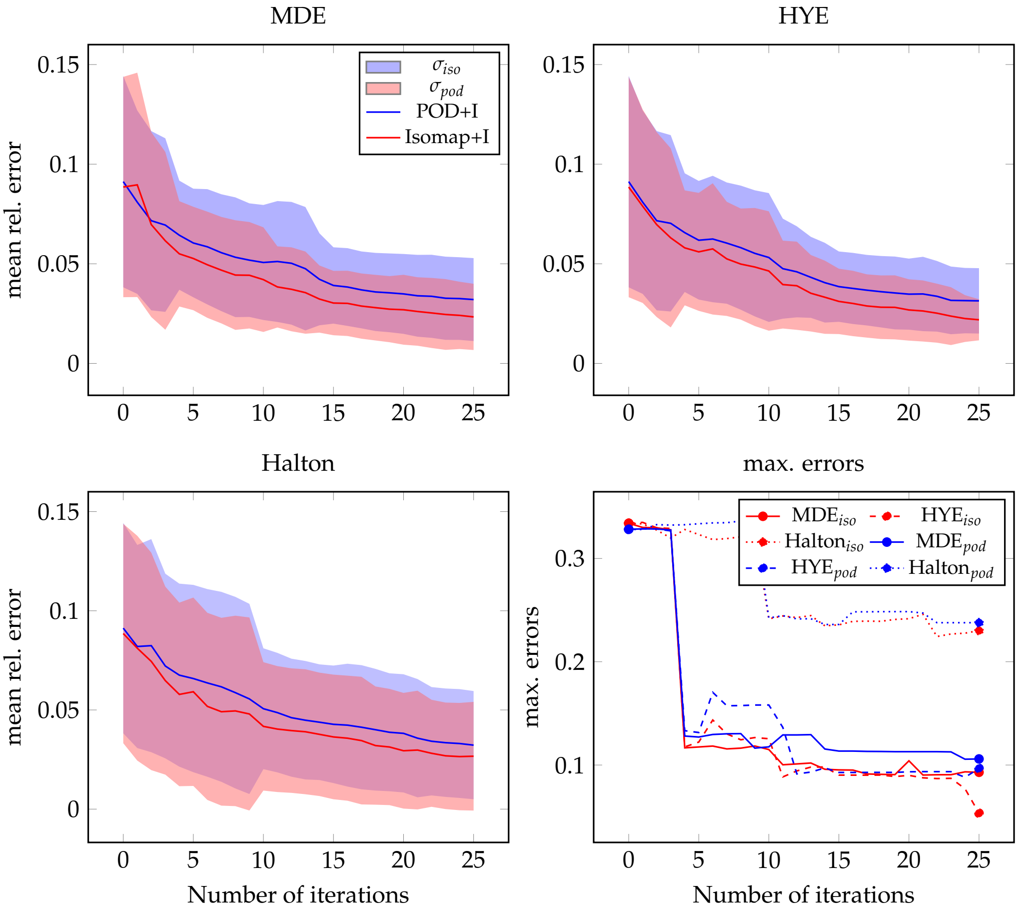


Figure 7.10. In the upper Figures and the bottom left Figure, the mean relative errors in predicting the surface C_p distribution by Isomap+I and POD+I along with the standard deviations σ are shown for various sampling strategies after each iteration. The maximal relative errors after each iteration are depicted in bottom right Figure.

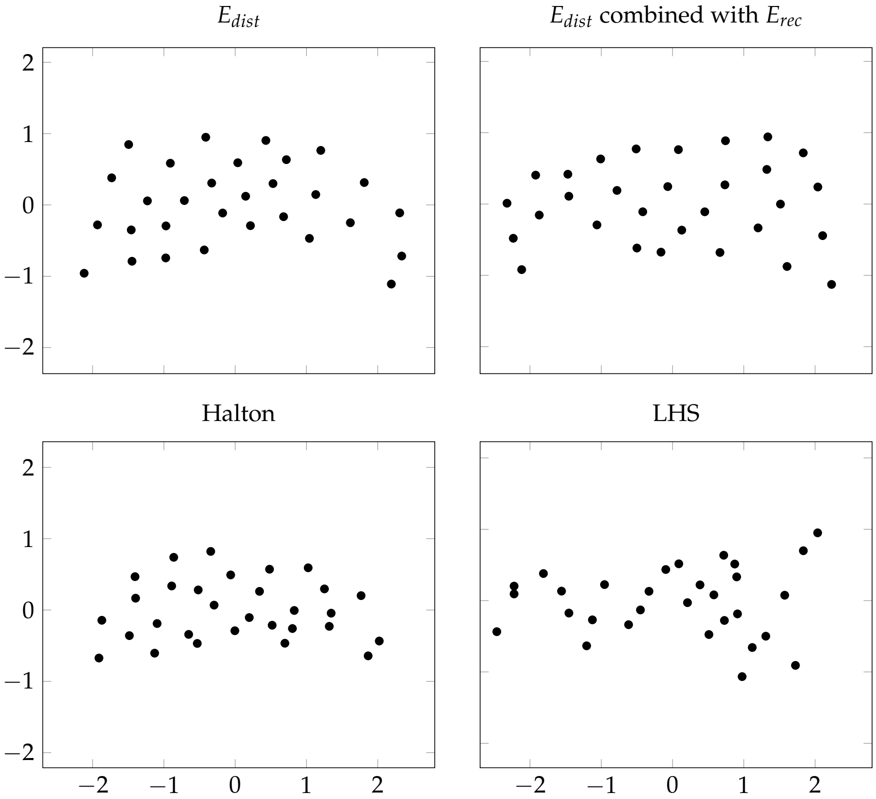
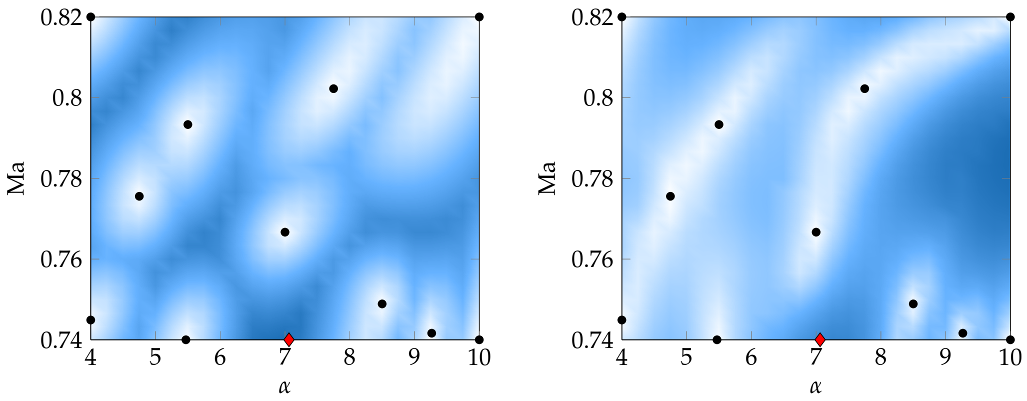
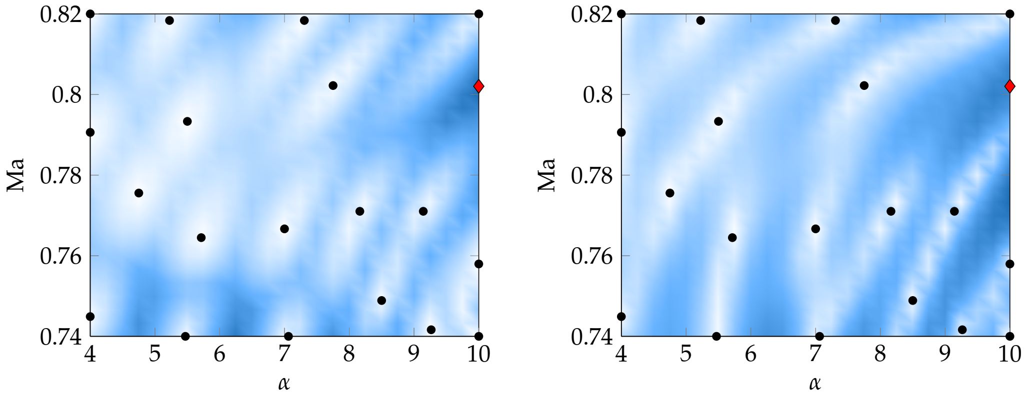


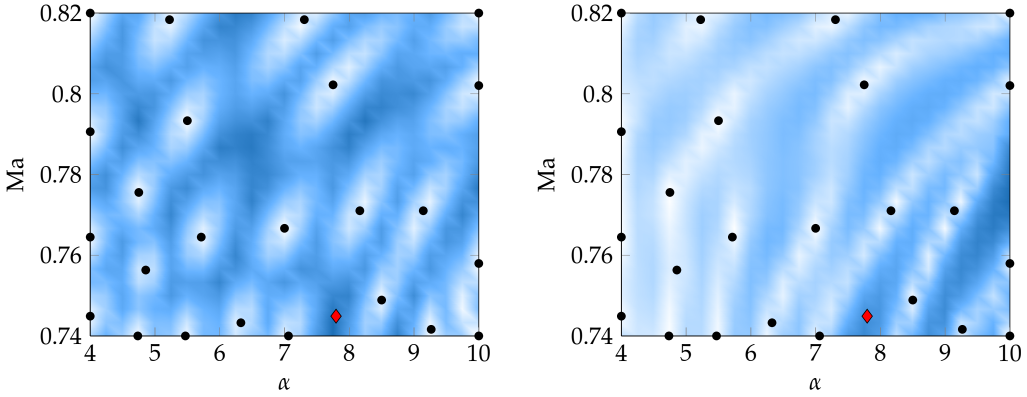
Figure 7.11. Embeddings of the final samplings (see Figure 7.9) obtained by various sampling methods and DoEs for the NACA 64A010 test case.



(a) E_{dist} error (left) and the actual error (right) after 6 iterations.

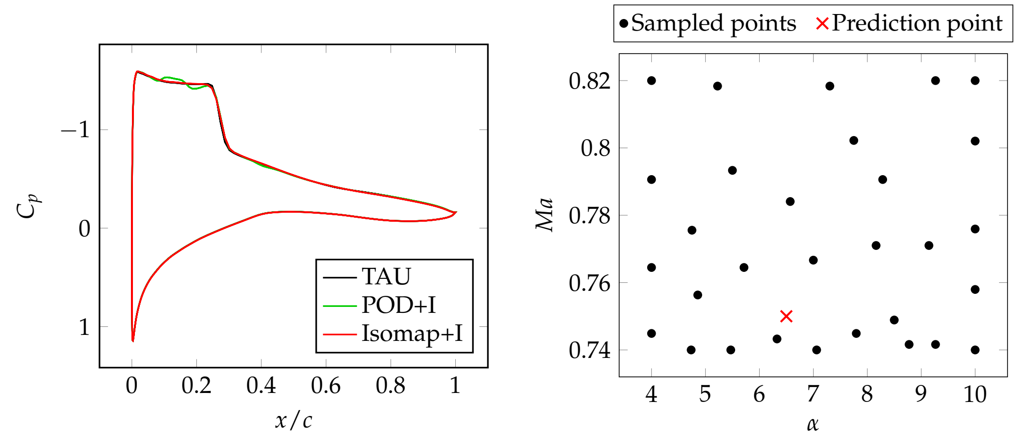


(b) E_{rec} error and the actual error after 14 iterations.



(c) E_{dist} error and the actual error after 19 iterations.

Figure 7.12. Errors during the sampling process of HYE. On the left side, the error plots of E_{dist} or E_{rec} are shown. The actual relative errors are depicted on the right side. The red diamond is the optimal parameter combination regarding E_{dist} or E_{rec}. For all plots it holds that the darker the blue the larger the error.



(a) Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (6.5°, 0.75)$.

(b) HYE based final sampling.

Figure 7.13. Prediction of the surface C_p-distribution at $(\alpha, \mathrm{Ma}) = (6.5°, 0.75)$ based on the sampled snapshots via HYE.

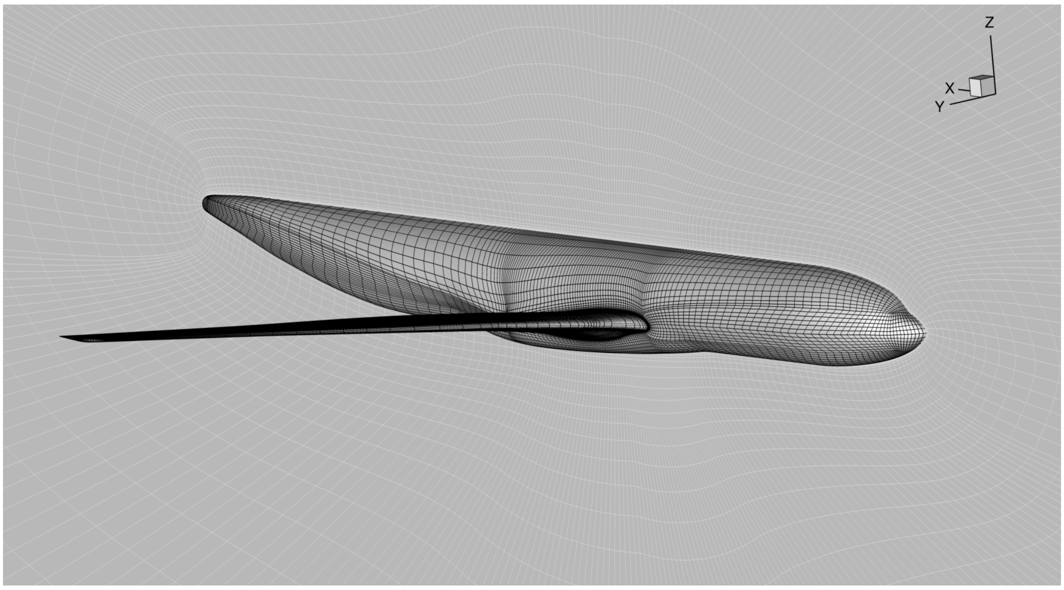


Figure 7.14. Detailed view of the surface of the computational structured grid of the XRF-1 configuration.

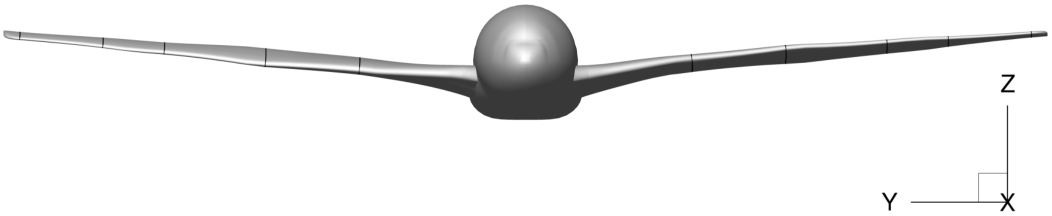


Figure 7.15. Front view of the XRF-1. Twists are performed at five section cuts depicted by the black lines on the wing. The left and the right wing show the maximum twists in different rotation directions.

Considering the results of Section 7.1.3, HYE is employed to generate a sampling of the data manifold $\mathcal{W}$ given by varying the five twist parameters of the configuration in the parameter space $\mathcal{P} = [-0.2, 0.2] \times [-2, 2] \times [-3, 3] \times [-2, 2] \times [-1, 1] \subset \mathbb{R}^5$, where the intervals from left to right correspond to the twist sections from fuselage to tip. HYE is performed based on $\tilde{m} = 20$ Halton samplings until $m = 100$ sample points are computed in total, i. e., 80 sampling iterations are accomplished. In iteration $1 \leq i \leq 80$, the corresponding viscous flow solution snapshot $\boldsymbol{W}^i = \boldsymbol{W}(\boldsymbol{p}^i)$, $\boldsymbol{p}^i = \boldsymbol{p}^* \in P_{\tilde{m}+i} \subset \mathcal{P}$, is computed with the TAU RANS solver (see Section 2.2) using the negative SA one-equation turbulence model [2], whereby the normalized density residual is reduced by six orders of magnitude for each solution. Since a target lift coefficient of $C_l = 0.5$ is aimed at, the angel of attack α varies during the CFD simulation until the target lift is matched.

As will be shown, even if the untried prediction points $\tilde{\boldsymbol{p}} \in \mathcal{P} \setminus P$ are far off the sampled data, the results are still accurate. Hence, for example, the Isomap based ROMs can be exploited for a multidisciplinary optimization within the whole parameter space, saving the costs of computing full-order CFD solutions.

The sampling process including the computation of the flow solutions and all further computations in Section 7.2.1 and Section 7.2.2 were performed in parallel on the DLR C^{2}A^{2}S^{2}E-2 cluster using one node endowed with 128 GB RAM and two Intel® Xeon® E5-2695 v2 Processors (30M Cache, 2.40 GHz, 12 Cores). Computing a full CFD solution for this test case until the density residual converged by six orders of magnitude took 5393 iterations or 4214 CPU seconds on average (based on the computed reference solutions of Section 7.2.1 and Section 7.2.2).

7.2.1. Isomap with interpolation

After conducting HYE, a data manifold $W = \{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^{100}\} \subset \mathcal{W}$ is obtained with corresponding parameter set $P = \{\boldsymbol{p}^1, \ldots, \boldsymbol{p}^{100}\} \subset \mathcal{P}$. The surface features 19,211 grid points, hence $\boldsymbol{W}^i \in \mathbb{R}^{19{,}211}$ for the interpolation based ROMs. Making use of the *a priori* information that five parameters are varied, the Isomap algorithm is applied to surface C_p-distribution vectors to compute a ($d = 5$)-dimensional embedding consisting of 100 representatives $\boldsymbol{y}^i \in \mathbb{R}^5$. To ensure an optimal embedding regarding the stress function, the number of nearest neighbors for constructing the neighborhood graph is automatically determined to be 87 (see Figure 7.16). Since the smallest stress values is obtained by considering almost every snapshot as a neighbor, i. e., the approximated geodesic distances, which barly differ from the Euclidean distances in the high-dimensional space in that case, are well-preserved in the low-dimensional space, it can be assumed that the data lies at least close to a five-dimensional linear subspace in the high-dimensional space. Therefore, the difference between the C_p predictions of the different ROMs is expected to be small. The adaptive method for choosing the number of nearest neighbors for the back-mapping is utilized with a minimum number of 10 neighbors and the maximum number of nearest neighbors is restricted to 20. The lower and upper bounds for the weights a_i are set to $\xi_l = -0.2$ and $\xi_u = 1.2$, respectively. Summarizing, Isomap+I is performed with the following parameters:

m	n	d	N	N_{rec}
100	19,211	5	87	10 – 20

The variables m, n, d, N and N_{rec} denote the number of snapshots, the dimension of the full-order solution space, the dimension of the embedding space, the number of nearest neighbors used to build the neighborhood graph and the number of nearest neighbors employed by the back-mapping, respectively.

For comparison purposes, a global POD of the 100 full-order surface C_p snapshots is performed, yielding a basis consisting of 99 orthonormal POD eigenmode vectors of dimension 19,211, since the mean of the snapshots is subtracted (see Section 3.1). As before, the POD model is combined with a TPS interpolation scheme as introduced in Section 3.3. Compared to Isomap+I, where a representative $\boldsymbol{y}^* \in \mathbb{R}^5$ of dimension five has to be interpolated to obtain a surface C_p prediction, POD+I employs TPS to interpolate the POD coefficient vector $\boldsymbol{a} \in \mathbb{R}^{99}$ of much larger dimension.

Isomap+I and POD+I were built in 119 and 0.17 CPU seconds, respectively, including the data processing, setting up the TPS model and, in the case of Isomap, the computation of the proper number of nearest neighbors and, due to the loss

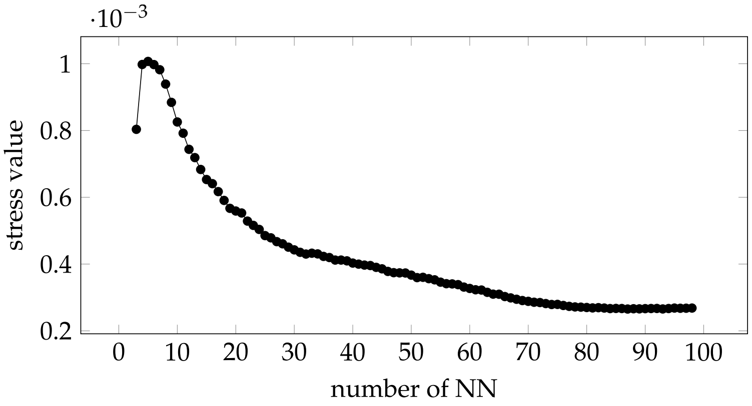


Figure 7.16. Stress function applied to the XRF-1 test case restricted to surface C_p data.

function, additional variable alternation. Although there is a big difference between the building times of Isomap+I and POD+I, compared to a full CFD calculation the offline times (without the snapshot computations) are negligible. The online prediction of a surface solution at an untried parameter combination $\tilde{\boldsymbol{p}} \in \mathcal{P} \setminus P$ took less than 0.01 CPU seconds for both ROMs, whereas a full CFD solution took 4214 CPU seconds on average. In other words, the predictions of both ROMs are more than 400,000 times faster than a full CFD solution, but certainly due to a trade-off of less accuracy.

The resulting surface C_p-distributions predicted by Isomap+I and POD+I for various parameter combinations $\tilde{\boldsymbol{p}} \in \mathcal{P} \setminus P$ are compared to the corresponding TAU reference solutions in Figures 7.17-7.20. The contour plot shows the reference surface C_p-distribution, whereby the white and black line denote the contour lines of the surface C_p predictions of Isomap+I and POD+I, respectively. The detailed surface C_p-distributions plotted against the normalized chord length x/c at four section cuts, ordered line by line from left to right corresponding to the twist sections from fuselage to tip, are depicted in the four line plots. To ensure that the prediction points vary significantly from each $\boldsymbol{p} \in P$, the untried prediction points $\tilde{\boldsymbol{p}} \in \mathcal{P} \setminus P$ were chosen as the centers of the 10 simplices with the largest volume of the *Delaunay triangulation* (see Section B) of P. The errors in terms of equation (7.1) are given in Table 7.5.

In Figure 7.17 and Figure 7.18, the best and worst prediction in terms of the error (7.1) are show, respectively. In Figure 7.17, the Isomap+I prediction matches the contour of the TAU reference solution quiet accurately. This is verified by the four section cuts depicting a detailed view of the surface C_p-distribution at different section cuts. The POD+I prediction also yields accurate predictions, but comparing the C_p-distribution between the first two sections, it differs from the reference solution.

The complexity of the test case is illustrated in Figure 7.18. Due to the arbitrary twists at five sections of the wing, two shocks arise on the upper side of the wing. Here, both methods yield good predictions for this complex shock shape, but, nevertheless, both ROMs fail to predict an accurate shock close to the wing tip. However, comparing the sections cuts and the contour plot, the Isomap+I prediction features at least slightly more accurate shocks than the POD+I prediction.

The two shocks also appear in the reference solution at $\tilde{\boldsymbol{p}}^{10}$ shown in Figure 7.19. Isomap+I and POD+I both fail to predict this complex shape, but, at least for Isomap+I, the surface predictions shown in the remaining line plots are reasonable. In Figure 7.20, Isomap+I yields an accurate prediction over most part of the configuration. Only close to the wing tip the prediction of the C_p-distribution is more inexact (cf. line plot on the buttom right of Figure 7.20). However, POD+I does not feature an accurate prediction of the C_p-distribution. Either the C_p prediction shows alternating discrepancies at the plateau upstream of the shock or the shock differs from the reference solution.

Moreover, the spanwise distributions of the partial force f_z and the partial moment m_y, which are calculated via AeroForce [90], are depicted in Figure 7.21, where the subscript specifies the component of the force or moment. In comparison to the spanwise distributions of the partial force f_z and the partial moment m_y of the reference solution, shown in Figure 7.21 and denoted by the black dotted line, the respective distributions of the ROM predictions differ a linewidth in the choosen resolution at most. Only at $\boldsymbol{p}^4$ an offset to the reference distributions is discernable.

As mentioned before, a target C_l of 0.5 is also prescribed in this test case. Hence the predictions of both ROMs should yield C_p-distributions with corresponding pressure components of C_l that match the pressure component of the C_l value of the reference solution. An overview of the pressure components of the reference C_l, the pressure components of the predicted C_l values and the corresponding relative errors are stated in Table 7.6. For boths ROMs the relative errors are less than $1.032 \cdot 10^{-3}$ and hence the discrepancy is less than $\sim 0.001\%$ for all predictions. Overall, however, the pressure components of C_l based on POD+I predictions are more accurate than the Isomap+I based values. This can partly be explained by the fact, that POD coupled with a RBF interpolation without additional polynomial term and calculating its integrated coefficients equals the direct interpolation of the integrated coefficients [94, Section 2.3].

$\tilde{\boldsymbol{p}}$	NN	Isomap+I	Isomap+LSQ	POD+I	POD+LSQ
1	11	$8.5930{\cdot}10^{-2}$	$7.1702{\cdot}10^{-2}$	$9.4556{\cdot}10^{-2}$	$5.9426{\cdot}10^{-2}$
2	16	$8.7347{\cdot}10^{-2}$	$7.5275{\cdot}10^{-2}$	$1.1523{\cdot}10^{-1}$	$5.5323{\cdot}10^{-2}$
3	14	$7.7126{\cdot}10^{-2}$	$7.5143{\cdot}10^{-2}$	$9.4977{\cdot}10^{-2}$	$6.3302{\cdot}10^{-2}$
4	10	$1.0089{\cdot}10^{-1}$	$1.3703{\cdot}10^{-1}$	$1.3514{\cdot}10^{-1}$	$6.4285{\cdot}10^{-2}$
5	20	$6.4210{\cdot}10^{-2}$	$7.1029{\cdot}10^{-2}$	$7.7599{\cdot}10^{-2}$	$5.7465{\cdot}10^{-2}$
6	10	$7.0431{\cdot}10^{-2}$	$9.3372{\cdot}10^{-2}$	$7.7246{\cdot}10^{-2}$	$5.9561{\cdot}10^{-2}$
7	14	$6.7638{\cdot}10^{-2}$	$5.1527{\cdot}10^{-2}$	$9.1839{\cdot}10^{-2}$	$4.4243{\cdot}10^{-2}$
8	10	$4.3445{\cdot}10^{-2}$	$4.5011{\cdot}10^{-2}$	$8.9193{\cdot}10^{-2}$	$3.0280{\cdot}10^{-2}$
9	13	$5.6208{\cdot}10^{-2}$	$4.0005{\cdot}10^{-2}$	$7.4482{\cdot}10^{-2}$	$2.8970{\cdot}10^{-2}$
10	10	$6.3993{\cdot}10^{-2}$	$5.9396{\cdot}10^{-2}$	$8.7417{\cdot}10^{-2}$	$4.1372{\cdot}10^{-2}$

Table 7.5. Errors in terms of equation (7.1) between the TAU reference surface C_p solutions and the surface C_p predictions obtained by Isomap+I, Isomap+LSQ, POD+I and POD+LSQ at various parameter combinations. The column NN lists the number of nearest neighbors employed by the Isomap based predictions at each parameter combination $\tilde{\boldsymbol{p}}$.

	Ref.	Isomap+I		POD+I	
$\tilde{\boldsymbol{p}}$	C_{l_p}	C_{l_p}	rel. err.	C_{l_p}	rel. err.
1	0.50015	0.50044	$5.785 \cdot 10^{-4}$	0.50004	$2.261 \cdot 10^{-4}$
2	0.50005	0.50036	$6.306 \cdot 10^{-4}$	0.50012	$1.365 \cdot 10^{-4}$
3	0.50001	0.50013	$2.403 \cdot 10^{-4}$	0.50015	$2.779 \cdot 10^{-4}$
4	0.50009	0.50040	$6.240 \cdot 10^{-4}$	0.50008	$2.347 \cdot 10^{-5}$
5	0.50001	0.50033	$6.439 \cdot 10^{-4}$	0.50026	$5.002 \cdot 10^{-4}$
6	0.50001	0.50027	$5.200 \cdot 10^{-4}$	0.50017	$3.232 \cdot 10^{-4}$
7	0.50011	0.50025	$2.817 \cdot 10^{-4}$	0.50005	$1.155 \cdot 10^{-4}$
8	0.50001	0.50024	$4.569 \cdot 10^{-4}$	0.50000	$8.355 \cdot 10^{-6}$
9	0.49999	0.50050	$1.032 \cdot 10^{-3}$	0.49992	$1.398 \cdot 10^{-4}$
10	0.50014	0.50035	$4.087 \cdot 10^{-4}$	0.49991	$4.796 \cdot 10^{-4}$

Table 7.6. Pressure components of C_l of the surface C_p predictions obtained by Isomap+I and POD+I at various parameter combinations and their relative errors regarding the pressure component of C_l of the reference solution.

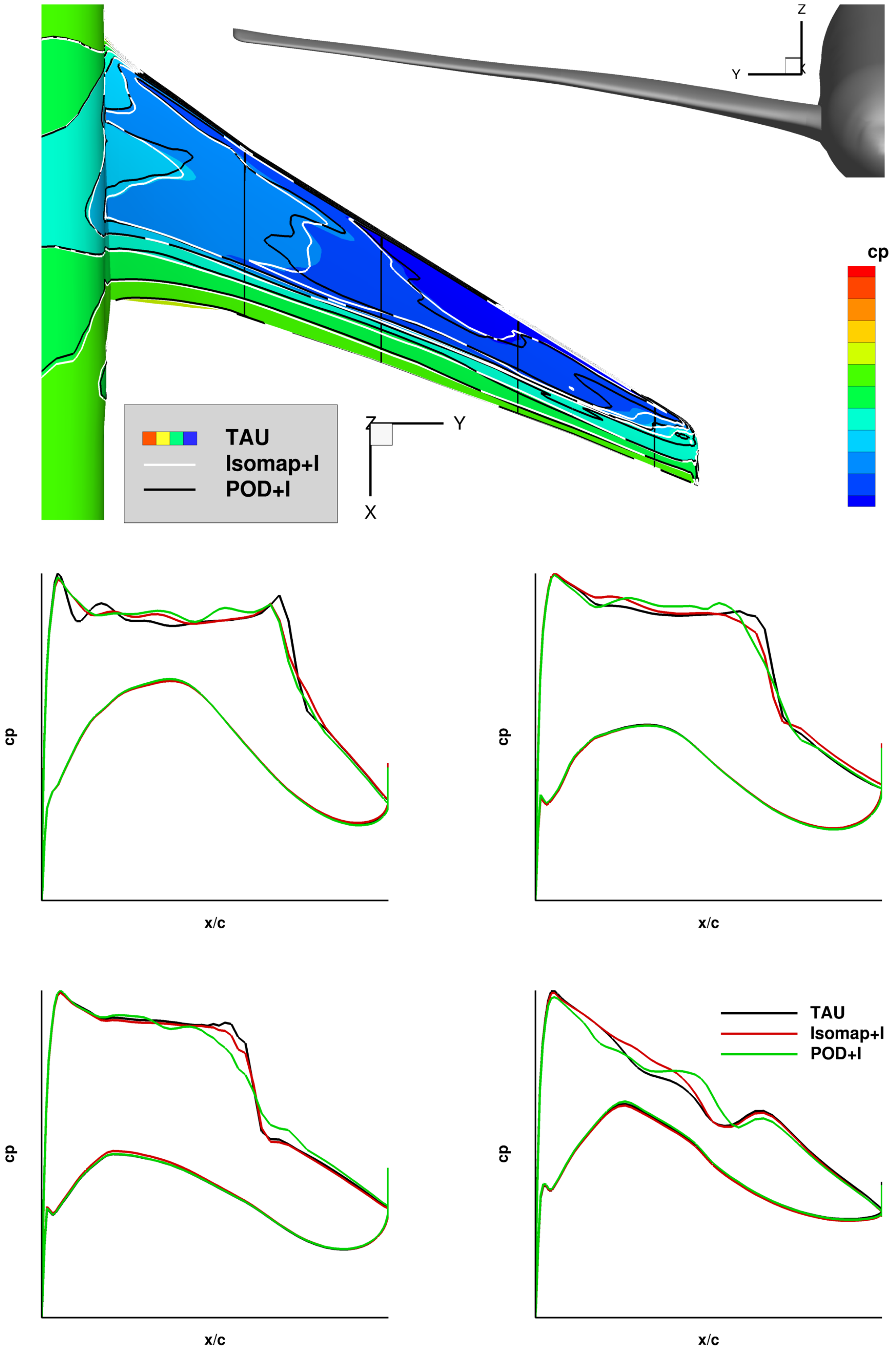


Figure 7.17. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^8 \in \mathcal{P} \setminus P$.

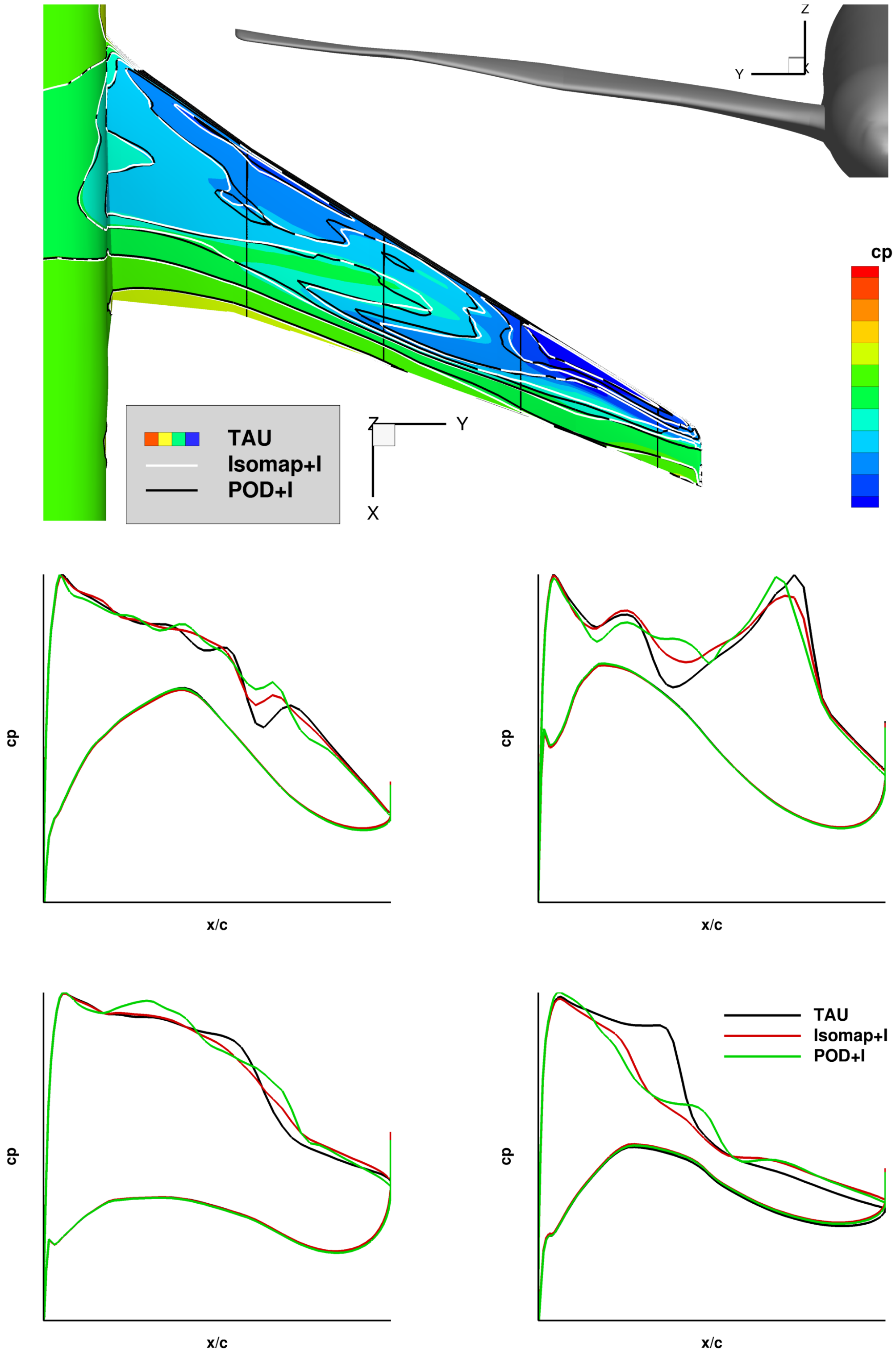


Figure 7.18. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^4 \in \mathcal{P} \setminus P$.

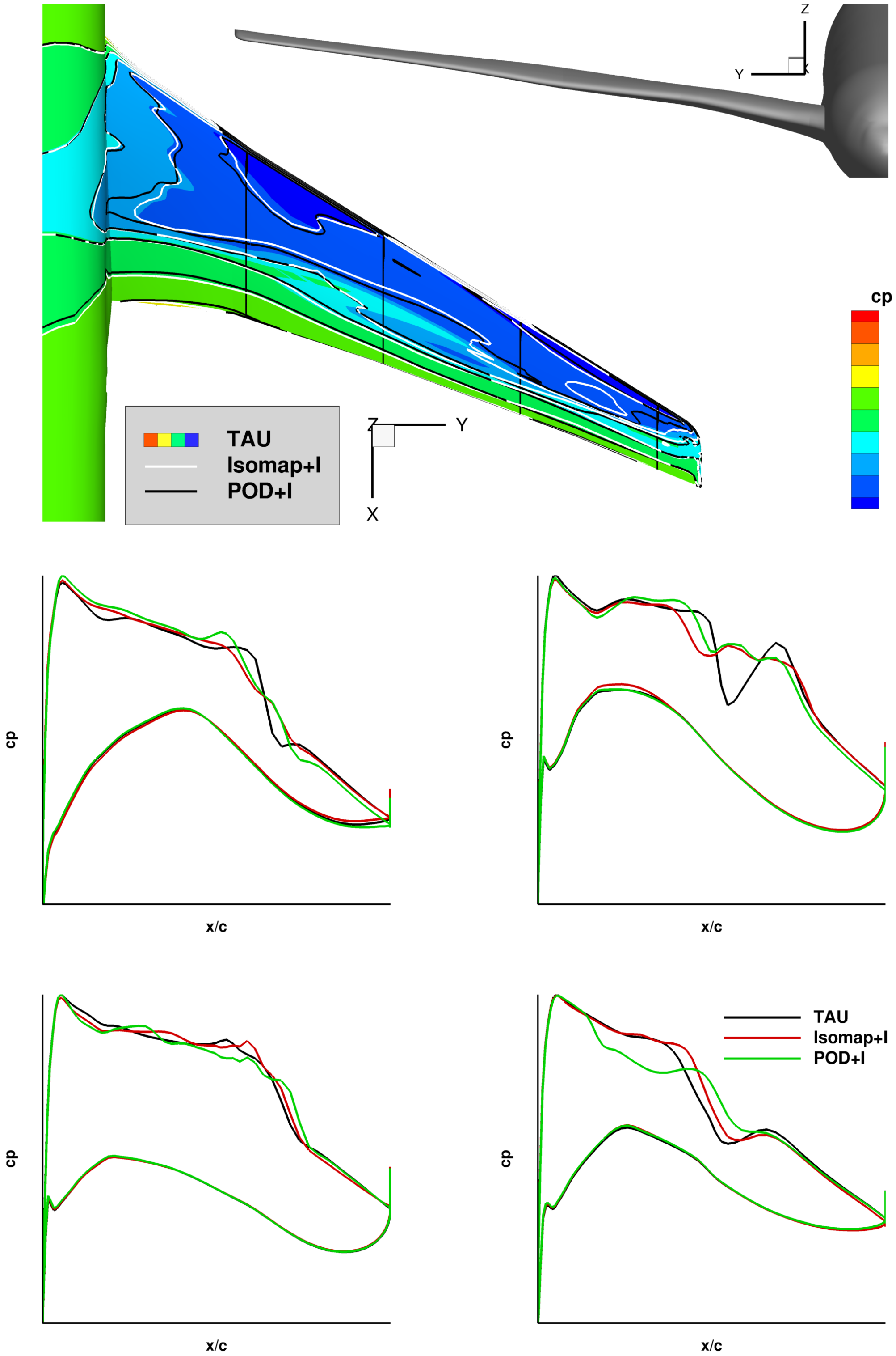


Figure 7.19. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^{10} \in \mathcal{P} \setminus P$.

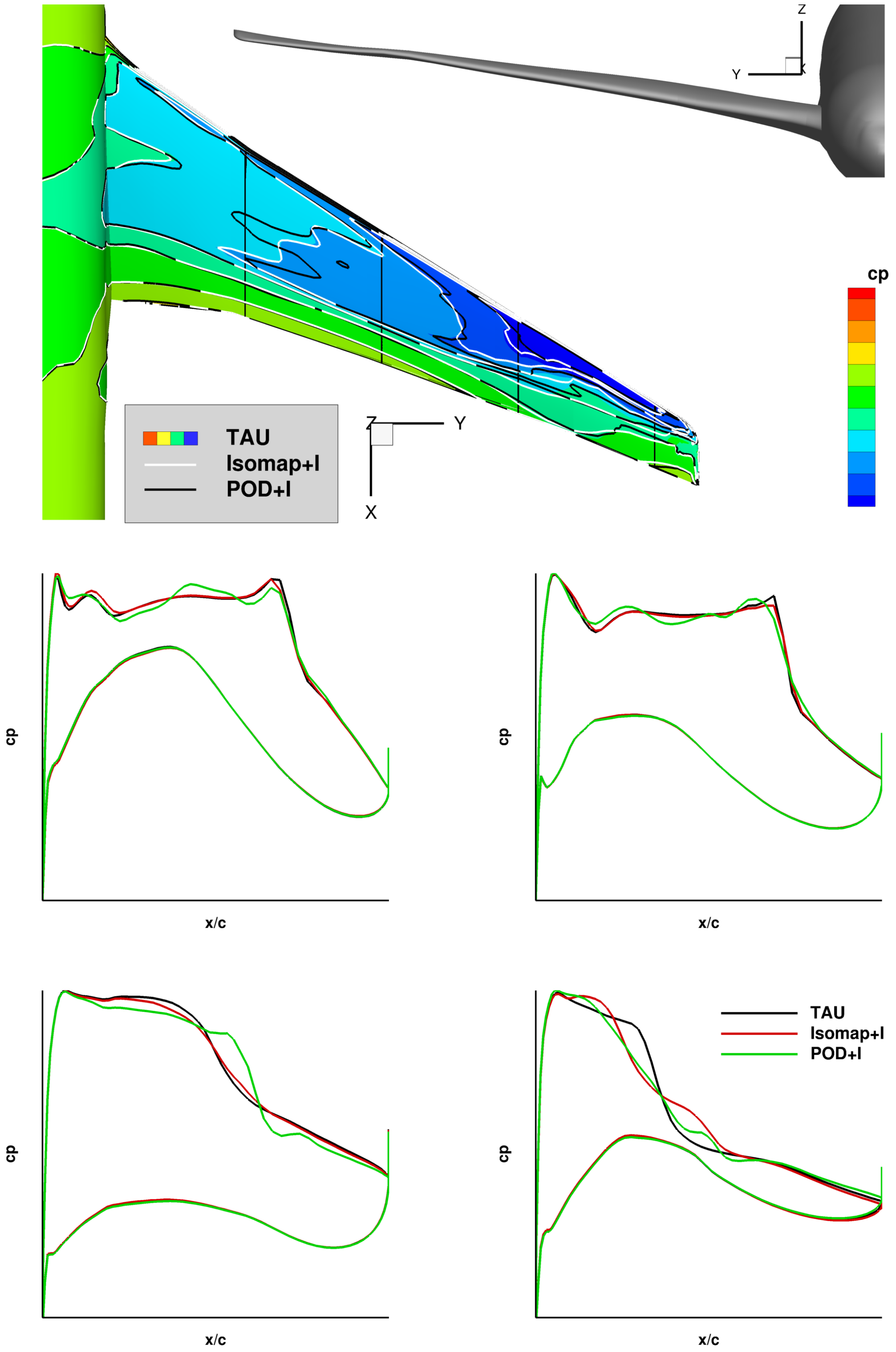
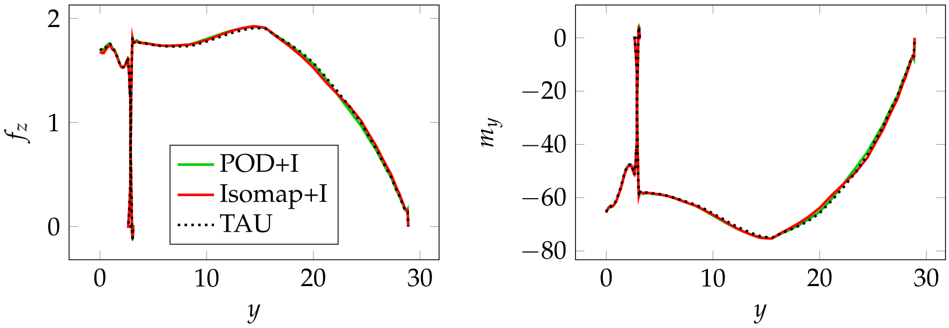
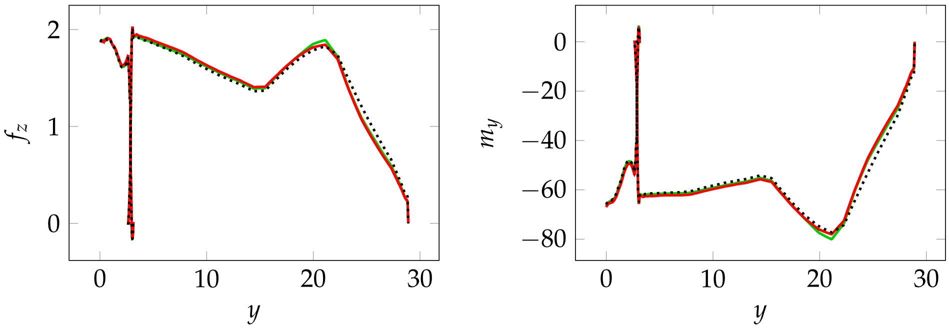


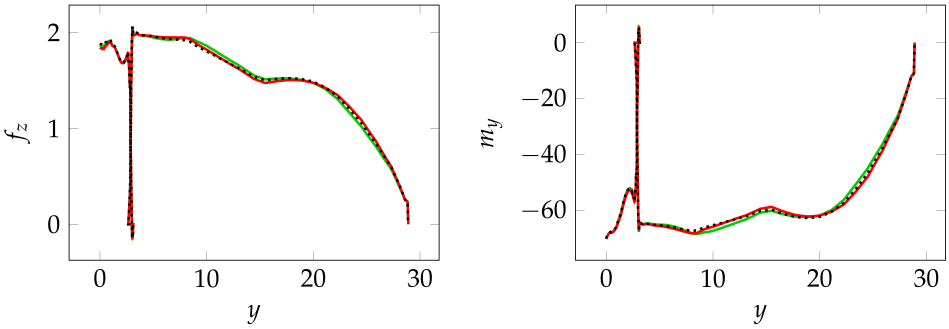
Figure 7.20. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^3 \in \mathcal{P} \setminus P$.



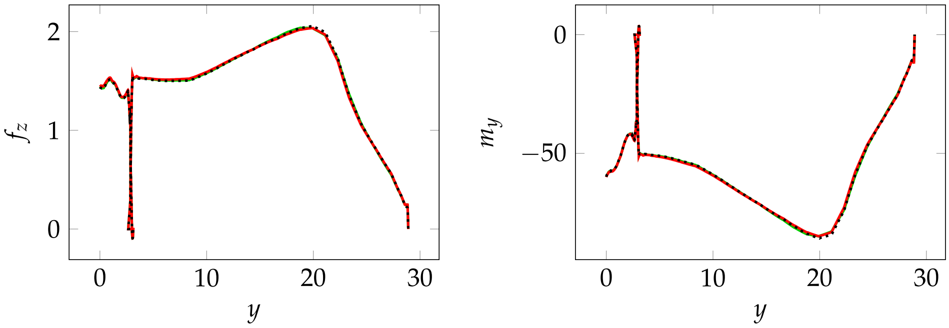
(a) Spanwise distributions at $\tilde{\boldsymbol{p}}^8$.



(b) Spanwise distributions at $\tilde{\boldsymbol{p}}^4$.



(c) Spanwise distributions at $\tilde{\boldsymbol{p}}^{10}$.



(d) Spanwise distributions at $\tilde{\boldsymbol{p}}^3$.

Figure 7.21. Spanwise distributions of f_z (left column) and m_y (right column) at various predictions points $\tilde{\boldsymbol{p}}$.

7.2.2. Isomap with residual optimization

The interpolation-based Isomap and POD coefficients are now exploited as starting values for the residual based ROMs derived in Sections 4.6 and 3.4. Unlike in Section 7.1.2, the unconstrained optimization problems are solved now, since the sequential least squares programming (SLSQP) [50] algorithm does not exploit the structure of the problem which would lead to large CPU times for this test case. To optimize the coefficients of an Isomap or POD based prediction, the Levenberg-Marquardt algorithm [60, 91] with additional Broyden's rank one updates of the Jacobian is applied to the unconstrained optimization problems (4.28a) and (3.25). This method is obtained by using the same approximation of the Hessian as in the Gauss-Newton method [91, Section 10.3], but instead of performing a line search, the Levenberg-Marquardt algorithm employs a trust-region strategy.

As in Section 7.1.2, all primitive variables plus the C_p-distribution are taken into account, leading to a set of snapshots $W \subset \mathbb{R}^n$, $n = n_g \cdot n_v = 6,275,072$. Hence, Isomap+LSQ and POD+LSQ are based on 100 snapshots of dimension $n = 6,275,072$, increasing the building time of the POD based ROM to 117 CPU seconds. The costs of Isomap+LSQ (119 CPU seconds) remains unaffected of the new data set as only the surface C_p-distribution is exploited to compute the embedding.

The residual has to be evaluated with proper boundary conditions Ma and α, which are not specified by the varied parameters as in Section 7.1.2. While the Mach number Ma is fixed, the angle of attack α varies for each flow solutions to ensure the specified target C_l value. To prescribe the proper boundary conditions for the residual evaluations, the angle of attack α could be interpolated based on the α-values of the snapshot set. However, α is obtained from the corresponding reference solution to exclude an additional error source, which would affect the accuracy of the C_p predictions. As before, after evaluating the TAU residual, the discrepancy in the total energy values belonging to the 20% smallest cells is exploited by the objective function to optimize the coefficient vector $\boldsymbol{a} \in \mathbb{R}^d$, until the tolerance of termination ($tol = 1.49012 \cdot 10^{-8}$) is reached. Here, d has the values $d \in [10, 20]$ or $d = 99$ for Isomap+LSQ or POD+LSQ, respectively.

The CFD enhanced predictions at various parameter combinations $\tilde{\boldsymbol{p}} \in \mathcal{P} \setminus P$ are compared to the corresponding TAU reference solutions in Figure 7.22 and Figure 7.23. The contour plot shows the reference surface C_p-distribution, whereby the white and black line denote the contour lines of the surface C_p predictions of Isomap+LSQ and POD+LSQ, respectively. The detailed surface C_p-distributions at four section cuts plotted against the normalized chord length x/c are depicted by the solid lines in the four line plots. The red dashed line and green dashed line show the initial solutions obtained by Isomap+I and POD+I, respectively. The line plots are ordered line by line from left to right corresponding to the twist sections from fuselage to tip. The spanwise distributions of the partial force f_z and the partial moment m_y are depicted in Figure 7.24. After conducting the residual optimization at $\tilde{\boldsymbol{p}}^7$, both ROMs show an improved prediction of the shock in the first two section cuts. In the last two cuts, both methods fall short on an accurate match of the C_p-distributions. However, both corresponding errors in terms of equation (7.1) are reduced (see Table 7.5) and the spanwise distributions depicted in Figure 7.24(a) match the reference distributions almost exactly. This also holds for the LSQ-based predictions at $\tilde{\boldsymbol{p}}^9$ depicted in Figure 7.23. The error is reduced (see Table 7.5) and improvements of the C_p predictions can be seen in the last two sections cuts. Furthermore, the spanwise distributions match the reference distributions almost exactly (see Figure 7.24(a)). Nevertheless, an enhancement of the complex shock shape depicted in the second section cut did not take place.

Considering that the Isomap+I predictions feature a smaller error than the POD+I predictions in advance, Isomap+LSQ increases the error at few parameter combinations $\tilde{\boldsymbol{p}}$ for this set up (see Table 7.5), unlike POD+LSQ. This may be avoided by imposing the boundary constraints (4.28) as in Section 7.1.2, but the arising CPU costs using SLSQP for this test case cannot be afforded. However, by applying Isomap+LSQ an improvement of the predictions is observed in most cases. Comparing the CPU times in Table 7.8, due to the less DoFs the Isomap+LSQ predictions are up to 4 times faster than the POD+LSQ predictions and, in average, 7.5 times faster than a full CFD computation.

7.3. Reduced-order models for aero-data for loads and structural sizing

In the context of multidisciplinary optimization (MDO) a CFD solver is repeatedly used to perform fluid/structure-coupled simulations. Typically, the entire optimization process consists of two nested loops: an inner loop, where different load cases are computed to size the different optimization regions of the structural model for a given aerodynamic shape, and an outer loop where performance data is computed and used to optimize the shape of the wing or aircraft according to some objective function. Since the computations of the full-order CFD solutions are expensive and repeatedly required in both loops, ROMs may provide remedy. For the outer loop, the steady aerodynamic ROMs introduced in Section 7.2.1 and Section 7.2.2 can be exploited to compute the necessary aerodynamic quantities for the optimization of

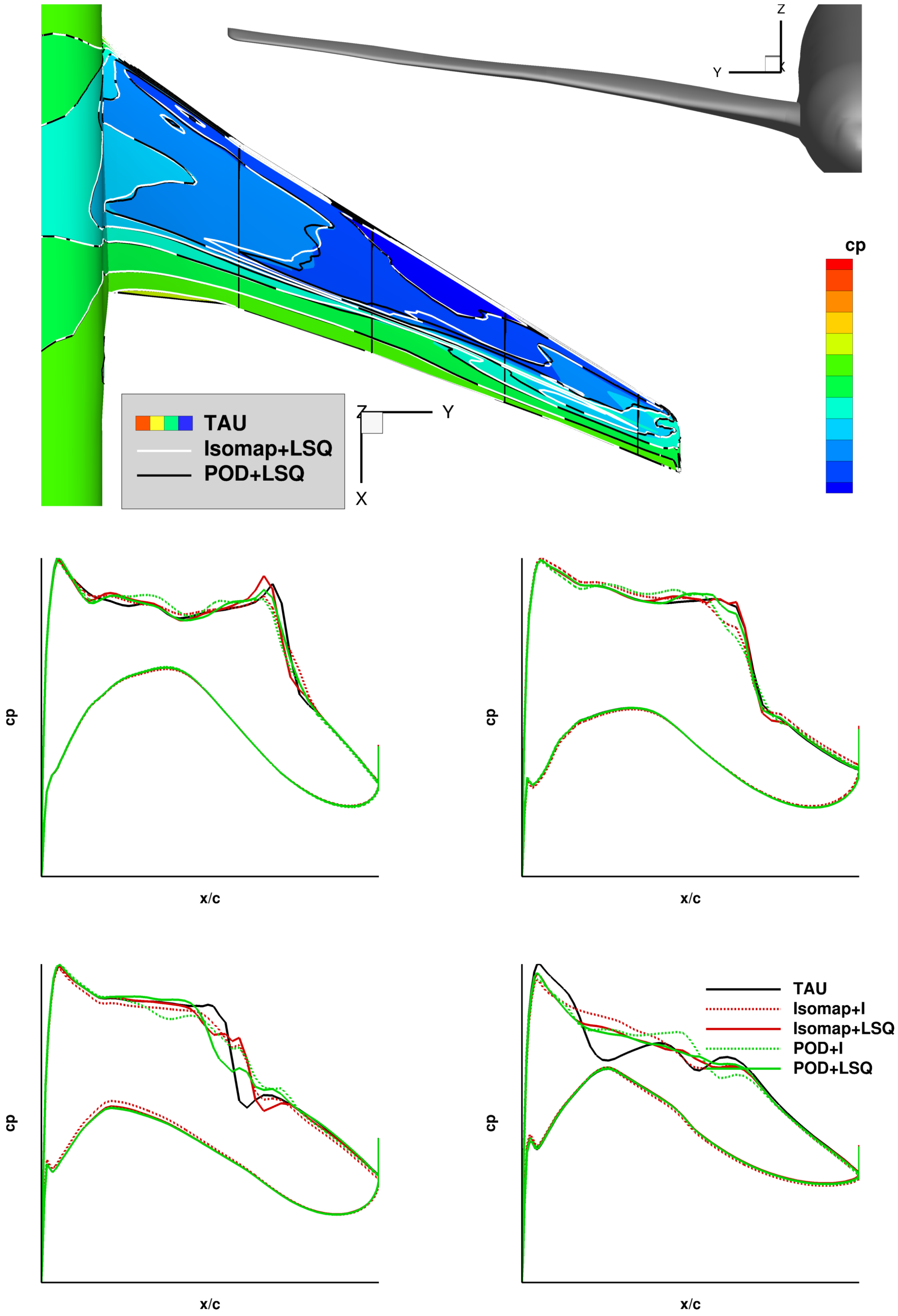


Figure 7.22. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^7 \in \mathcal{P} \setminus P$.

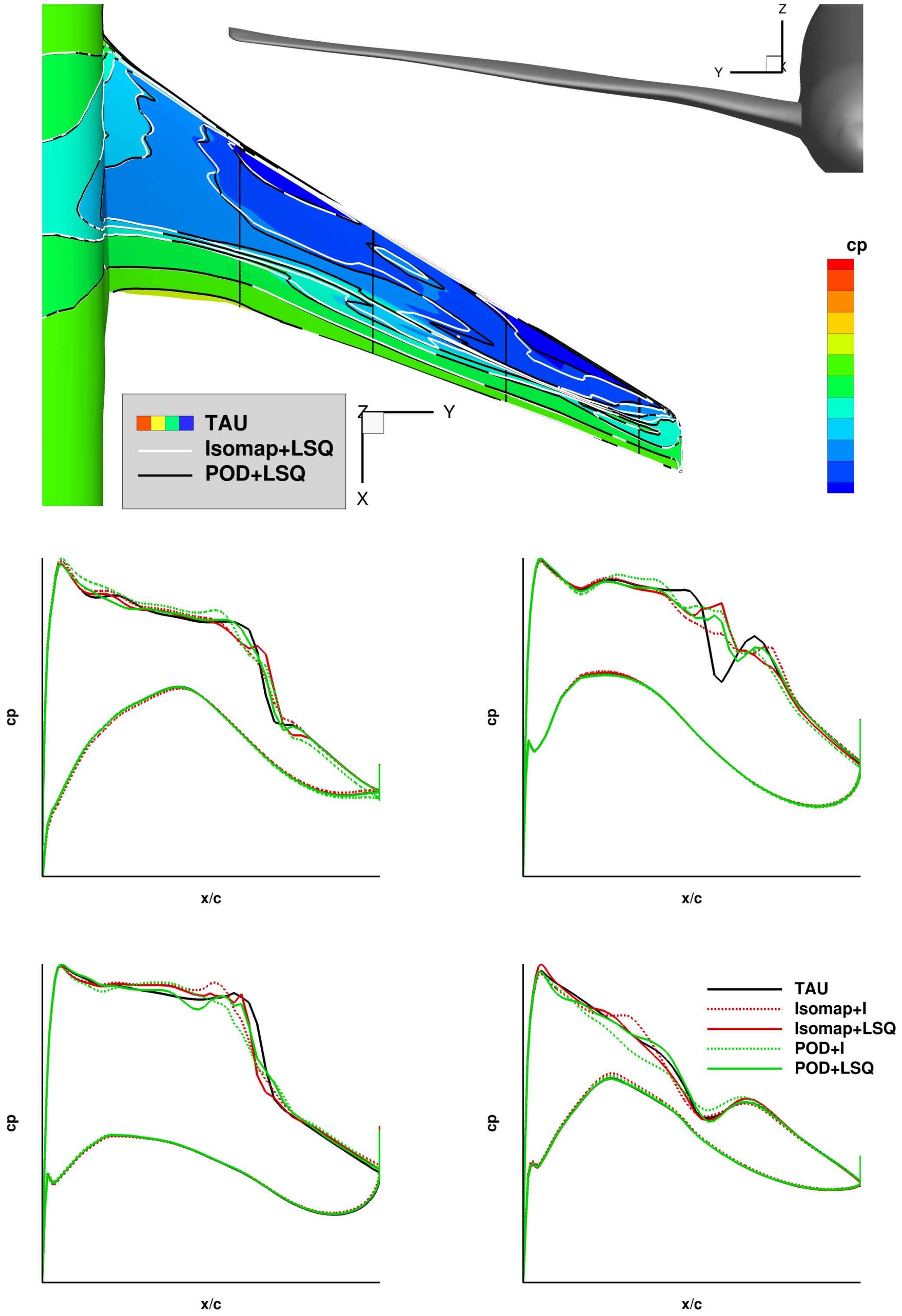
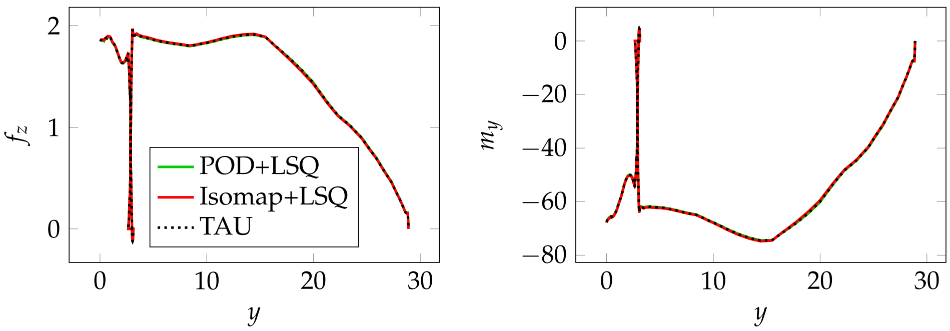


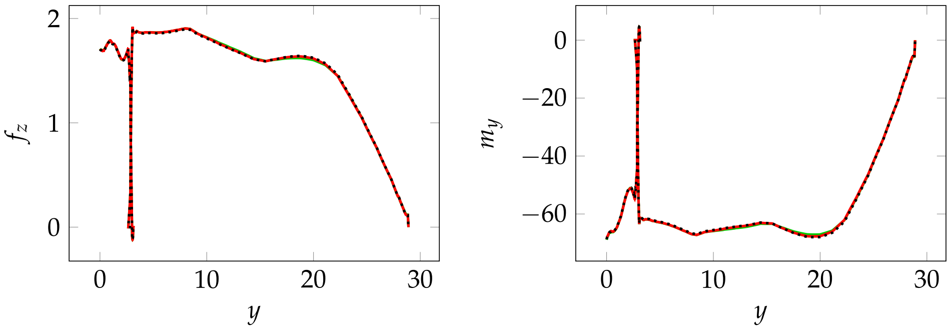
Figure 7.23. Prediction of the surface C_p-distribution of the XRF-1 by Isomap+I and POD+I at an untried parameter combination $\tilde{\boldsymbol{p}}^9 \in \mathcal{P} \setminus P$.

	Ref.	Isomap+LSQ		POD+LSQ	
$\tilde{\boldsymbol{p}}$	C_{l_p}	C_{l_p}	rel. err.	C_{l_p}	rel. err.
1	0.50015	0.50030	$2.895 \cdot 10^{-4}$	0.50010	$9.418 \cdot 10^{-5}$
2	0.50005	0.50026	$4.340 \cdot 10^{-4}$	0.50006	$2.466 \cdot 10^{-5}$
3	0.50001	0.50014	$2.483 \cdot 10^{-4}$	0.50003	$4.048 \cdot 10^{-5}$
4	0.50009	0.50054	$9.151 \cdot 10^{-4}$	0.50004	$8.473 \cdot 10^{-5}$
5	0.50001	0.50023	$4.404 \cdot 10^{-4}$	0.50004	$5.440 \cdot 10^{-5}$
6	0.50001	0.50011	$1.924 \cdot 10^{-4}$	0.50012	$2.177 \cdot 10^{-4}$
7	0.50011	0.50020	$1.799 \cdot 10^{-4}$	0.50008	$4.771 \cdot 10^{-5}$
8	0.50001	0.50026	$5.114 \cdot 10^{-4}$	0.50009	$1.592 \cdot 10^{-4}$
9	0.49999	0.50015	$3.273 \cdot 10^{-4}$	0.50005	$1.316 \cdot 10^{-4}$
10	0.50014	0.50041	$5.264 \cdot 10^{-4}$	0.50009	$1.022 \cdot 10^{-4}$

Table 7.7. Pressure components of C_l of the surface C_p predictions obtained by Isomap+LSQ and POD+LSQ at various parameter combinations and their relative errors regarding the pressure component of C_l of the reference solution.



(a) Spanwise distributions at $\tilde{\boldsymbol{p}}^7$.



(b) Spanwise distributions at $\tilde{\boldsymbol{p}}^9$.

Figure 7.24. Spanwise distributions of f_z (left column) and m_y (right column) at various predictions points $\tilde{\boldsymbol{p}}$.

$\tilde{p}$	Isomap+LSQ iter	Isomap+LSQ CPU times (s)	POD+LSQ iter	POD+LSQ CPU times (s)	speed-up factor
1	23	568.39	19	1,905.21	3.35
2	27	714.16	19	1,900.94	2.66
3	19	557.12	17	1,873.29	3.36
4	22	540.33	19	1,905.38	3.53
5	21	682.94	21	1,920.56	2.81
6	18	479.97	19	1,904.7	3.97
7	19	554.35	14	1,816.53	3.28
8	22	539.72	19	1,897.45	3.52
9	18	523.81	16	1,849.99	3.53
10	18	480.95	15	1,829.26	3.8

Table 7.8. Number of iterations and CPU times of conducting Isomap+LSQ and POD+LSQ predictions for the XRF-1 test case. The gained speed-up by performing Isomap+LSQ instead of POD+LSQ is stated in the last column.

the geometry, since the variation of the geometry is taken into account. The advantage of substituting the CFD solver with a ROM is that the snapshots and the ROM are computed offline before the optimization takes place. This should lead to a speed-up of the actual optimization process or rather to a larger amount of load cases that can be considered for the structural sizing in the inner loop.

To demonstrate the idea it is assumed here that the aircraft is rigid, i.e., there is no need to perform fluid/structure-coupled simulations. This is true for some very stiff models, e.g., for wind-tunnel models. A rigid aircraft may also be assumed when computing CFD-based corrections to a linear potential flow solution in the context of loads computations (so-called AIC corrections). Furthermore, it is assumed that a single (steady) load case is sufficient to size the structural model. This is also a valid simplification in some case because sizing the structure for the most critical load case may result in a structural mass that is close to the result of sizing the structure for several (but less critical) load cases. Hence, skipping the inner loop, the ROMs for the XRF-1 are exploited to predict the steady surface C_p-distributions for different geometries, which afterwards are fed into the structural sizing process to obtain the skin thickness of each optimization region.

For the XRF-1 test case, the structural optimization regions are divided into upper skin regions, lower skin regions, wing ribs regions and wing spar regions. The skin thicknesses at the wing spar, which is the main structural member of the wing, are shown in Figures 7.25-7.26 for various parameters $\boldsymbol{p} \in \mathcal{P} \setminus P$. The three spars are divided in the optimization regions from 234 to 287, 288 to 303 and 304 to 347. Since

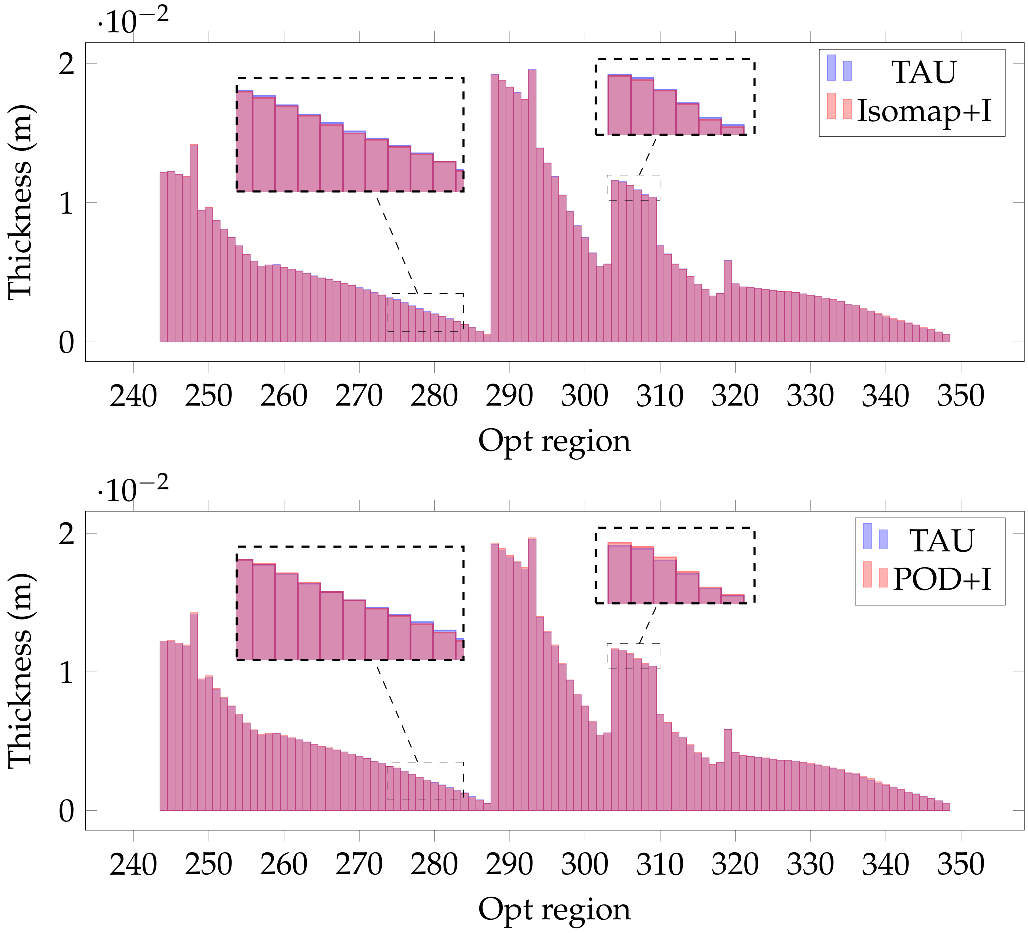


Figure 7.25. Thicknesses at the wing spar optimization regions for the predictions of Isomap+I (top) and POD+I (bottom) at $\boldsymbol{p}^3$. Emphasized are the regions with largest gap.

there are three wing spars, the thickness distribution plotted against the optimization region does not show a decreasing behaviour as usual, but for each separate spar it does. However, close to the optimization region 320, there is an outlier which may be due to the fact that the inner spar, with corresponding optimization regions 288 to 303, ends and the loads are distributed to the two remaining spars. As it can be seen, Isomap and POD-based ROMs provide good predictions of the thickness distributions at the wing spar optimization regions. Particularly, the detailed views emphasize that there is almost no mismatch between the computed thicknesses of the predicted solutions and the computed thicknesses of the corresponding reference solutions. Thus, these ROMs are suitable in the context of MDO and should lead to a speed-up of the optimization process as mentioned above.

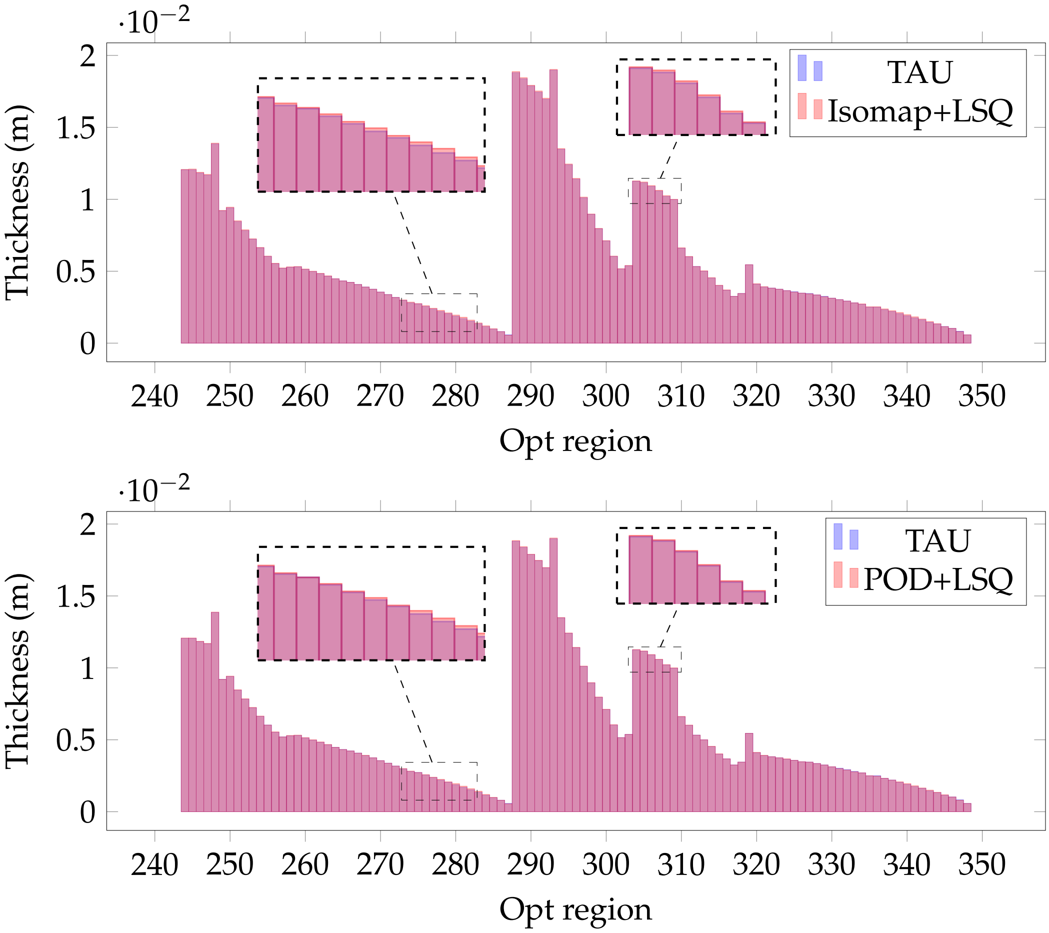


Figure 7.26. Thicknesses at the wing spar optimization regions for the predictions of Isomap+LSQ (top) and POD+LSQ (bottom) at $\boldsymbol{p}^7$. Emphasized are the regions with largest gap.

CHAPTER 8

Conclusions

In this thesis, an efficient DR method for capturing the geometry of the manifold of a set of CFD solutions was introduced and applied to steady transonic flow problems. "By coupling Isomap with an interpolation method and by using a suitable back-mapping, a reduced-order model, called Isomap+I, has been developed and was successfully applied to 2D and 3D aerodynamic test cases. Instead of making use of the underlying physics directly, it relies on the fact that the flow physics were well captured by the snapshots and hence by the geometry of the manifold, which is then exploited by Isomap to obtain solutions for new parameters. As such, the Isomap+I method proposed in this work is tailored for parametric steady flow problems and does not readily apply to unsteady flows. Isomap+I was compared to POD combined with interpolation (POD+I). Even though Isomap+I eventually leads to a linear combination of precomputed snapshot solutions, the featured results indicate that the (pseudo-) isometric snapshot neighborhood selection and the weighted back projection considered here may prevent or at least alleviate the step-function behavior frequently observed for the POD+I approach at the shock location. It was shown in particular that the surface pressure distributions predicted by Isomap+I are more accurate than the results obtained by POD+I, especially in terms of the shock representation. [...] Just like POD+I, the Isomap+I predictions do not require a projection of the governing equations on some reduced basis but a collection of sampled CFD snapshots. Furthermore, an arbitrary set and number of flow variables suffices to build an Isomap+I ROM, while projection-based ROMs require all flow variables to be considered."(Franz et al. [36])

By exploiting the flux residual of the underlying equations, the coefficients of the Isomap+I predictions has been enhancend, leading to an improved prediction of linear combination of precomputed snapshot solutions. This method, called Isomap+LSQ, was also successfully applied to 2D and 3D aerodynamic test cases and compared to POD combined with residual optimization (POD+LSQ). Since only a small number of nearest neighbors is used for an Isomap based prediction, the DoF of the corresponding optimization problem is much smaller than the DoF of an POD based optimization problem, where the DoF equals the number of employed modes. Thus, the costs of performing Isomap+LSQ is much cheaper than performing POD+LSQ. A POD with mode reduction may compensate this drawback, but, however, the residual optimization just slightly improved the initial solution, which was still far off the reference solution compared to the predictions of the remaining ROMs. Although the DoF is smaller for Isomap+LSQ, the accuracy of the predictions matches up to the accuracy of POD+LSQ predictions as the snapshots with the proper characteristics are exploited.

Due to the automated selection of a proper neighborhood size for the neighborhood graph and the adaptive choice of the number of nearest neighbors employed by the back-mapping, which ensure a proper set of the essential metaparameters of the Isomap based ROMs, two user-friendly ROMs are provided. Moreover, Isomap, or more precisely, MDS itself was enhanced by another loss function, which adapts the dimensionality reduction to our purposes, more precisely, a loss function focussing on a more accurate preservation of the distances of close-by snapshots.

The fact, that the predictions of both Isomap based ROMs depend only on a small set of neighboring snapshots, reduces the impact of the curse of dimensionality when the dimension of the parameter space is increasing. With growing dimension, the number of snapshots increases dramatically in order to cover all characteristics appearing in the solutions throughout the parameter space. However, the number of nearest neighbors employed for the back-mapping is independent of the dimension of parameter space, whereas the number of POD modes equals the number of snapshots and thus grows with the dimension of the parameter space.

To generally address the curse of dimensionalty, two adaptive sampling strategies, called MDE and HYE, have been developed. Both methods try to generate an evenly sampled data manifold leading to a better description of the underlying manifold and eventually to more accurate ROMs. HYE also considers the reconstruction error of an Isomap+I ROM during the sampling process, so that the sample density in the highly nonlinear regions of the manifold, where the error is expected to be larger, is enhanced. Both strategies were applied to 2D and 3D aerodynamic test cases. In particular, the advantages over random samplings and full factorial designs for Isomap based ROMs have been demonstrated based on the 2D NACA 64A010 test case. Furthermore, it was shown that employing HYE, 100 snapshots are sufficient to obtain global ROMs with satisfying accuracy within the 5-dimensional parameter space specified for the XRF-1 test case.

APPENDIX A

Radial basis functions model

Let a noise free data sampling $X = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^m\} \subset \mathbb{R}^k$ yielding the responses $Y = \{y^1, \dots, y^m\} \subset \mathbb{R}$ with a functional coherence $f(\boldsymbol{x}^i) = y^i$, $i = 1, \dots, m$, be given. The radial basis functions (RBF) model approximates the function f by chosing a basis $\{\psi_1, \dots, \psi_{m_c}\}$ of an m_c-dimensional linear function space $\mathcal{A}$ to seek an interpolant of the form

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}) = \sum_{i=1}^{m_c} w_i \psi(\|\boldsymbol{x} - \boldsymbol{c}^i\|), \tag{A.1}$$

where $\boldsymbol{\psi}(\boldsymbol{x}) = (\psi(\|\boldsymbol{x} - \boldsymbol{c}^1\|), \dots, \psi(\|\boldsymbol{x} - \boldsymbol{c}^{m_c}\|)$ is a vector containing the values of the basis functions $\psi_i = \psi(\|\boldsymbol{x} - \boldsymbol{c}^i\|)$ and $\boldsymbol{c}^i$ denotes the ith center of the m_c basis function centres. A potential choice of the basis function is the thin plate spline (TPS) $\psi(r) = r^2 \ln r$ or the Gaussian function $\psi(r) = e^{-r^2/(2\sigma^2)}$.

To determine the unknown weights $\boldsymbol{w} = (w_1, \dots, w_{m_c})$ the interpolation conditions

$$\hat{f}(\boldsymbol{x}^j) = \sum_{i=1}^{m} w_i \psi(\|\boldsymbol{x}^j - \boldsymbol{c}^i\|) = y^j, \quad j = 1, \dots, m, \tag{A.2}$$

can be exploited, which form a linear system of equations in terms of $\boldsymbol{w}$. Choosing the number of basis functions equal to the number of data samples with inner centres $\boldsymbol{x}^i$, i.e. $m_c = m$ and $\boldsymbol{c}^i = \boldsymbol{x}^i$, the systems simplifies to

$$\Psi \boldsymbol{w} = \boldsymbol{y}, \tag{A.3}$$

where Ψ is the so called *Gram matrix* with entries $\Psi_{i,j} = \psi(\|\boldsymbol{x}^i - \boldsymbol{x}^j\|)$, $i, j = 1, \dots, m$ and the responses are given as the vector $\boldsymbol{y} = (y_1, \dots, y_m)$. If the matrix Ψ is regular, the solution of (A.3) is given by

$$\boldsymbol{w} = \Psi^{-1} \boldsymbol{y}. \tag{A.4}$$

However, the matrix Ψ can be singular for sets of distinct centres for the TPS and the linear space $\mathcal{A}$ of the general form of RBF does not include either constant functions or polynomials of degree p, which are required in many applications. Remedy is given by adding a polynomial of degree $p \geq 1$ to the definition (A.1). Let $\{\varphi_1, \ldots, \varphi_{\hat{p}}\}$ be a basis of the space of polynomials of degree at most p denoted by Π_p, then the interpolant (A.1) is replaced by

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}) + \varphi(\boldsymbol{x}) = \sum_{i=1}^{m} w_i \psi(\|\boldsymbol{x} - \boldsymbol{x}^i\|) + \sum_{i=1}^{\hat{p}} \alpha_i \varphi_i(\boldsymbol{x}), \tag{A.5}$$

where $\varphi \in \Pi_p$. The additional *degrees of freedom (DoFs)* are eliminated by imposing the constraints

$$\sum_{i=1}^{m} w_i \varphi_j(\boldsymbol{x}^i) = 0, \quad j = 1, \ldots, \hat{p}. \tag{A.6}$$

Hence, a nonsingular system of linear equations of order $(m + \hat{p}) \times (m + \hat{p})$ has to be solved to determine the unknown coefficients w_i, $i = 1, \ldots, m$, and α_j, $j = 1, \ldots, \hat{p}$:

$$\begin{pmatrix} \Psi & P^T \\ P & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{w} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{pmatrix}, \tag{A.7}$$

where $P = (\varphi_j(x^i))_{ij} \in \mathbb{R}^{(m+\hat{p})}$, $i = 1, \ldots, m$, $j = 1, \ldots, \hat{p}$, and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{\hat{p}})^T$. A more detailed description of radial basis functions is given in [65] and [33].

APPENDIX B

Delaunay triangulation

B.1 Definition ([66])**.** A *triangle* $\Delta \subset \mathbb{R}^d$ is a d-dimensional simplex, which is defined by its $d+1$ vertices.

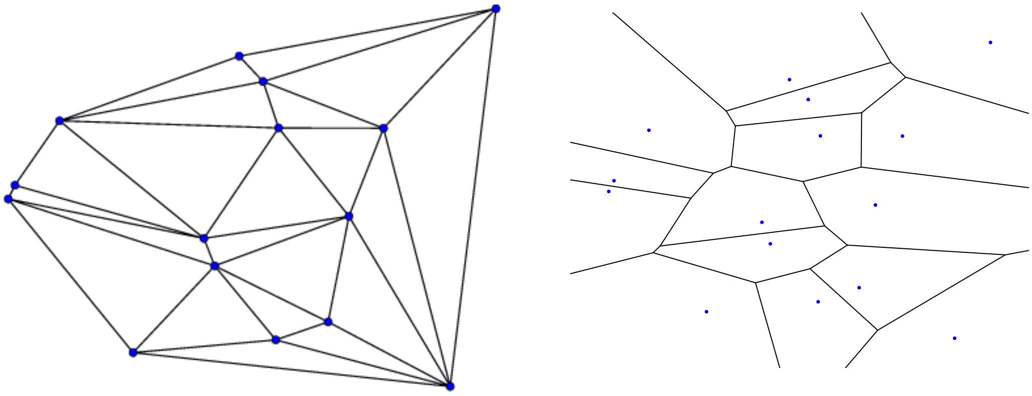
B.2 Definition ([66])**.** A *triangulation* $\mathcal{T}(P)$ of a set of points $P \subset \mathbb{R}^d$ is a simplicial decomposition of the convex hull of P where the vertices of the triangles are contained in P.

B.3 Definition ([66])**.** The *Delaunay triangulation* $\mathcal{DT}(P)$ of a set of points $P \subset \mathbb{R}^d$ is defined to be the triangulation such that the circumsphere of every triangle Δ in the triangulation contains no point $p \in P$ in its interior.

Note that the Delaunay triangulation is the dual of a Voronoi diagram and exist for every point set $P \subset \mathbb{R}^d$ [30]. Furthermore, the problem of finding the Delaunay triangulation of a set of points $P \subset \mathbb{R}^d$ can be rewritten as the problem of finding the convex hull of a set of transformed points $P' \subset \mathbb{R}^{d+1}$ [66, 10]. Hence, convex hull algorithms like *Quickhull*[1] [10] can be applied to compute a Delaunay triangulation. An example of a Delaunay triangulation and its dual is depicted in Figure B.1.

---

[1] `http://www.qhull.org/`



(a) Delaunay triangulation.

(b) Voronoi diagram.

Figure B.1. Delaunay triangulation and its dual Voronoi diagram for a set of random points.

Bibliography

[1] A. Al-Dmour and K. Mohammad. Active control of flexible structures using principal component analysis in the time domain. *Journal of sound and vibration*, 253(3):545–569, 2002.

[2] S. R. Allmaras and F. T. Johnson. Modifications and clarifications for the implementation of the spalart-allmaras turbulence model. In *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, pages 1–11, 2012.

[3] D. Alonso, J. Vega, and A. Velazquez. Reduced-order model for viscous aerodynamic flow past an airfoil. *AIAA journal*, 48(9):1946–1958, 2010.

[4] D. Amsallem, M. Zahr, and C. Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 2012.

[5] D. Amsallem, C. Farhat, and M. Zahr. On the robustness of residual minimization for constructing pod-based reduced-order cfd models. In *Proceedings of the 43rd AIAA Fluid Dynamics Conference and Exhibit*, San Diego, CA, 2013.

[6] C. Andrews, J. Davies, and G. Schwarz. Adaptive data compression. *Proceedings of the IEEE*, 55(3):267–277, 1967.

[7] P. Astrid. *Reduction of process simulation models: a proper orthogonal decomposition approach*. PhD thesis, 2004.

[8] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing point estimation in models described by proper orthogonal decomposition. *Automatic Control, IEEE Transactions on*, 53(10):2237–2251, 2008.

[9] M. Balasubramanian, E. L. Schwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford. The isomap algorithm and topological stability. *Science*, 295(5552): 7–7, 2002.

[10] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.

[11] A. G. Barnston and C. F. Ropelewski. Prediction of enso episodes using canonical correlation analysis. *Journal of Climate*, 5(11):1316–1345, 1992.

[12] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[13] T. Bechtold, M. Striebel, K. Mohaghegh, and E. ter Maten. Nonlinear model order reduction in nanoelectronics: Combination of pod and tpwl. *PAMM*, 8(1): 10057–10060, 2008.

[14] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.

[15] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[16] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[17] A. V. Bernstein and A. P. Kuleshov. Tangent bundle manifold learning via grassmann & stiefel eigenmaps. *CoRR*, abs/1212.6031, 2012.

[18] M. Bernstein, V. De Silva, J. C. Langford, and J. B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Technical report, Department of Psychology, Stanford University, 2000.

[19] J. Blazek. *Computational Fluid Dynamics: Principles and Applications*. Elsevier, Amsterdam – London – New York – Oxford – Paris – Shannon – Tokyo, 1st edition, 2001.

[20] M. Brand. Charting a manifold. In *Advances in neural information processing systems*, pages 961–968, 2002.

[21] T. Bui-Thanh, M. Damodaran, and K. Willcox. Proper orthogonal decomposition extensions for parametric applications in transonic aerodynamics. In *Proceedings of the 21th AIAA Applied Aerodynamics Conference*, 2003.

[22] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The gnat method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647, 2013.

[23] L. Cayton. Algorithms for manifold learning. *Univ. of California at San Diego Tech. Rep*, pages 1–17, 2005.

[24] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, 78(7):808–817, 2000.

[25] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[26] J. De Leeuw and P. Mair. Multidimensional scaling using majorization: Smacof in r. *Department of Statistics, UCLA*, 2011.

[27] V. de Silva and J. B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in neural information processing systems*, 15: 705–712, 2003.

[28] V. de Silva and J. B. Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, Technical report, Stanford University, 2004.

[29] Q. Du and M. D. Gunzburger. Centroidal voronoi tessellation based proper orthogonal decomposition analysis. *Control and estimation of distributed parameter systems*, pages 137–150, 2003.

[30] H. Edelsbrunner. *Algorithms in combinatorial geometry*, volume 10. Springer, 1987.

[31] J. Eftang, A. Patera, and E. Rønquist. An *"hp"* certified reduced basis method for parametrized elliptic partial differential equations. *SIAM Journal on Scientific Computing*, 32(6):3170–3200, 2010. doi: 10.1137/090780122.

[32] R. W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

[33] A. Forrester, A. Sóbester, and A. Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[34] M. Fossati and W. G. Habashi. Multiparameter analysis of aero-icing problems using proper orthogonal decomposition and multidimensional interpolation. *AIAA journal*, 51(4):946–960, 2013.

[35] R. Franke. A critical comparison of some methods for interpolation of scattered data. Technical report, DTIC Document, 1979.

[36] T. Franz, R. Zimmermann, S. Görtz, and N. Karcher. Interpolation-based reduced-order modelling for steady transonic flows via manifold learning. *International Journal of Computational Fluid Dynamics*, 0(0):1–16, 2014.

[37] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic press, 1990.

[38] S. Gallot, D. Hulin, and J. Lafontaine. *Riemannian Geometry*. Springer, 2004.

[39] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2012.

[40] J. C. Gower. Properties of euclidean and non-euclidean distance matrices. *Linear Algebra and its Applications*, 67:81–97, 1985.

[41] D. Gratton and K. Willcox. Reduced-order, trajectory piecewise-linear models for nonlinear computational fluid dynamics. In *34 th AIAA Fluid Dynamics Conference and Exhibit*, 2004.

[42] B. Haasdonk and M. Ohlberger. Reduced basis method for finite volume approximations of parametrized linear evolution equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 42(02):277–302, 2008.

[43] J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1):84–90, 1960.

[44] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[45] P. Holmes, J. L. Lumley, and G. Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 1998.

[46] A. Iske. Radial basis functions: basics, advanced topics and meshfree methods for transport problems. *Rendiconti del Seminario Matematico*, 61(3):247–285, 2003.

[47] K. Karhunen. *Zur Spektraltheorie stochastischer Prozesse*. Suomalainen tiedeakatemia, 1946.

[48] H. Klein. *Differenzierbare Mannigfaltigkeiten (lecture notes)*. Christian-Albrechts-Universität zu Kiel, 2002.

[49] D. Kosambi. Statistics in function space. *J. Indian Math. Soc*, 7(1):76–88, 1943.

[50] D. Kraft et al. *A software package for sequential quadratic programming*. DFVLR Obersfaffeuhofen, Germany, 1988.

[51] S. Langer, A. Schwöppe, and N. Kroll. The dlr flow solver tau-status and recent algorithmic developments. In *52nd AIAA Aerospace Sciences Meeting. AIAA Paper*, volume 80, 2014.

[52] M. H. Law and A. K. Jain. Incremental nonlinear dimensionality reduction by manifold learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(3):377–391, 2006.

[53] P. A. LeGresley and J. J. Alonso. Investigation of non-linear projection for pod based reduced order models for aerodynamics. *AIAA paper*, 926:2001, 2001.

[54] L. Lorente, J. Vega, and A. Velazquez. Generation of aerodynamic databases using high-order singular value decomposition. *Journal of Aircraft*, 45(5):1779, 2008.

[55] D. J. Lucia, P. S. Beran, and W. A. Silva. Reduced-order modeling: new approaches for computational physics. *Progress in Aerospace Sciences*, 40(1):51–117, 2004.

[56] H. V. Ly and H. T. Tran. Modeling and control of physical processes using proper orthogonal decomposition. *Mathematical and computer modelling*, 33(1):223–236, 2001.

[57] S. Maneewongvatana and D. M. Mount. It's okay to be skinny, if your friends are fat. In *Center for Geometric Computing 4th Annual Workshop on Computational Geometry*, volume 2, 1999.

[58] K. Mardia, J. Kent, and J. Bibby. Multivariate analysis. 1980.

[59] M. Mifsud. Reduced-order modelling for high-speed aerial weapon aerodynamics. 2008.

[60] J. J. Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[61] H. Narayanan and S. Mitter. Sample complexity of testing the manifold hypothesis. In *Advances in Neural Information Processing Systems*, pages 1786–1794, 2010.

[62] J. A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[63] B. Peherstorfer, D. Butnaru, K. Willcox, and H. Bungartz. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing*, 36(1): A168–A192, 2014.

[64] R. Pinnau. Model reduction via proper orthogonal decomposition. In W. Schilders, H. van der Vorst, and J. Rommes, editors, *Model Order Reduction: Theory, Research Aspects and Applications*, volume 13 of *Mathematics in Industry*, pages 95–109. Springer Berlin Heidelberg, 2008.

[65] M. J. D. Powell. Radial basis function methods for interpolation to functions of many variables. In *HERCMA*, pages 2–24, 2001.

[66] V. Rajan. Optimality of the delaunay triangulation in $\mathbb{R}^d$. *Discrete & Computational Geometry*, 12(1):189–202, 1994.

[67] M. Rewienski. *A trajectory piecewise-linear approach to model order reduction of nonlinear dynamical systems*. PhD thesis, Citeseer, 2003.

[68] M. Rewienski and J. White. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 22(2):155–170, 2003.

[69] M. Rewienski and J. White. Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations. *Linear algebra and its applications*, 415(2):426–454, 2006.

[70] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[71] O. Samko, A. D. Marshall, and P. L. Rosin. Selection of the optimal parameter value for the isomap algorithm. *Pattern Recognition Letters*, 27(9):968–979, 2006.

[72] J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 18(5):401–409, 1969.

[73] L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *The Journal of Machine Learning Research*, 4:119–155, 2003.

[74] W. H. Schilders, H. A. Van Der Vorst, and J. Rommes. *Model order reduction: theory, research aspects and applications*, volume 13. Springer, 2008.

[75] D. Schwamborn, T. Gerhold, and R. Heinrich. The DLR TAU-code: recent applications in research and industry, tech. report. In *European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2006*, The Netherlands, Sept. 2006. Egmond and Zee.

[76] C. Shao and H. Hu. Extension of isomap for imperfect manifolds. *Journal of Computers*, 7(7), 2012.

[77] C. Shao, H. Huang, and C. Wan. Selection of the suitable neighborhood size for the isomap algorithm. In *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pages 300–305. IEEE, Aug 2007.

[78] L. Shi, Q. Yang, Y. Xu, and P. He. A model of selecting the parameters based on the variance of distance ratios for manifold learning algorithms. In *Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on*, volume 2, pages 507–512. IEEE, 2009.

[79] L.-K. Shi and P.-L. He. An evaluated model based on the variance of distance ratios for nonlinear dimensionality reduction algorithms. In *Machine Learning and Cybernetics, 2007 International Conference on*, volume 2, pages 1144–1148. IEEE, 2007.

[80] J. Shlens. A tutorial on principal component analysis. *Systems Neurobiology Laboratory, University of California at San Diego*, 2005.

[81] L. Sirovich. Turbulence and the dynamics of coherent structures. i-coherent structures. ii-symmetries and transformations. iii-dynamics and scaling. *Quarterly of applied mathematics*, 45:561–571, 1987.

[82] M. Spivak. A comprehensive introduction to differential geometry. 1, 3ed, 1999.

[83] S. S. Stahara, A. J. Crisalli, and J. R. Spreiter. Evaluation of a strained coordinate perturbation procedure: Nonlinear subsonic and transonic flows. *AIAA Aerospace Sciences Meeting*, 18, 1980.

[84] Y. W. Teh and S. T. Roweis. Automatic alignment of local representations. In *Advances in neural information processing systems*, pages 841–848, 2002.

[85] J. Tenenbaum, V. De Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[86] C. Tropea, A. Yarin, and J. Foss. Springer Handbook of Experimental Fluid Mechanics. *Springer-Verlag Berlin Heidelberg*, 2007.

[87] M. W. Trosset, B. Hendrickson, C. kwong Li, and R. Mathias. Extensions of classical multidimensional scaling: Computational theory. *Computational Statistics*, 17:2002, 2002.

[88] L. Van der Maaten, E. Postma, and H. Van den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:1–41, 2009.

[89] K. Veroy, C. Prud'homme, D. Rovas, and A. Patera. A posteriori error bounds for reduced-basis approximation of parametrized noncoercive and nonlinear elliptic partial differential equations. In *Proceedings of the 16th AIAA computational fluid dynamics conference*, volume 3847, 2003.

[90] J. Wild. Aeroforce - thrust/drag bookkeeping and aerodynamic force breakdown over components. Technical report, German Aerospace Center (DLR), 1999.

[91] S. Wright and J. Nocedal. *Numerical optimization*, volume 2. Springer New York, 1999.

[92] R. Zimmermann. Towards best-practice guidelines for POD-based reduced order modeling of transonic flows. In C. Poloni, D. Quagliarella, J. Périaux, N. Gauger, and K. Giannakoglou, editors, *Proceedings of EUROGEN 2011*, Capua, Italy, September 2011. CIRA - Italian Aerospace Research Centre.

[93] R. Zimmermann. A locally parametrized reduced-order model for the linear frequency domain approach to time-accurate computational fluid dynamics. *SIAM Journal on Scientific Computing*, 36(3):B508–B537, 2014.

[94] R. Zimmermann and S. Görtz. Improved extrapolation of steady turbulent aerodynamics using a non-linear POD-based reduced order model. *The Aeronautical Journal*, 116(1184), Oct. 2012.

[95] R. Zimmermann, A. Vendl, and S. Görtz. Reduced-order modeling of steady flows subject to aerodynamic constraints. *AIAA Journal*, pages 1–12, 2014.